**UNIT-IV**

**Syllabus**:   
**User Search Techniques:** Search statements and binding, Similarity measures and  
ranking, Relevance feedback, Selective dissemination of information search, weighted  
searches of Boolean systems, Searching the Internet and hypertext.  
**Information Visualization:** Introduction, Cognition and perception, Information  
visualization technologies

***Search Statements and Binding:***

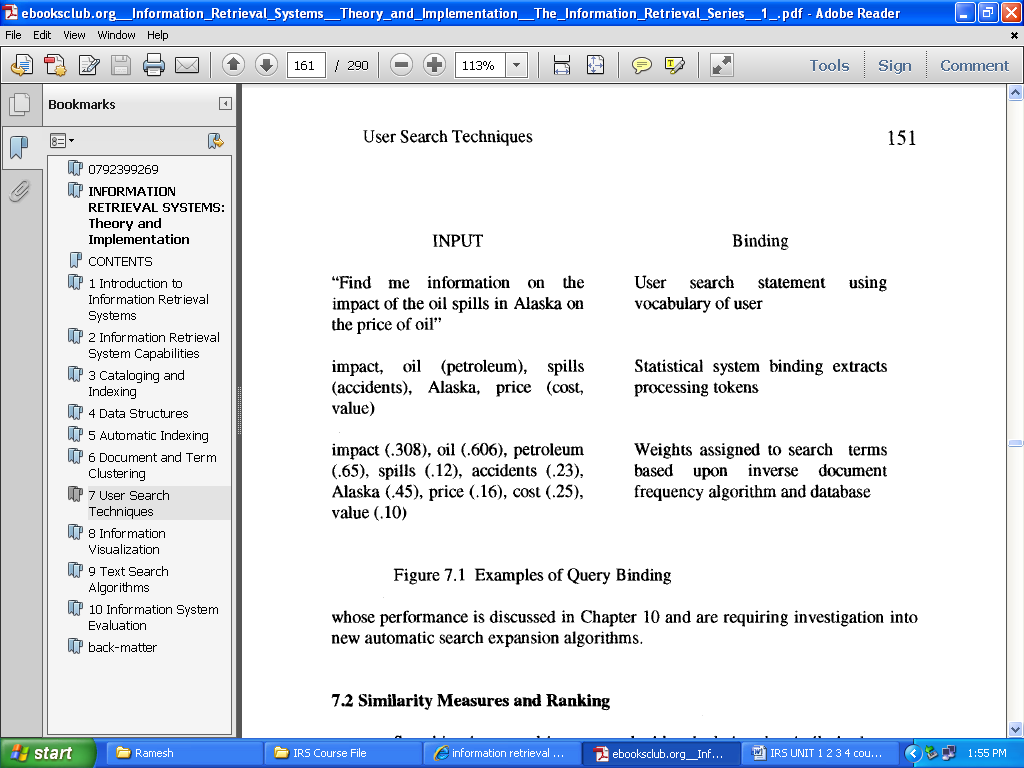
Search statements are the statements of an information need generated by users to specify the concepts they are trying to locate in items. In generation of the search statement, the user may have the ability to weight (assign an importance) to different concepts in the statement. At this point the binding is to the vocabulary and past experiences of the user.

**Binding** in this sense is when a more abstract form is redefined into a more specific form. The search statement is the user's attempt to specify the conditions needed to subset logically the total item space to that cluster of items that contains the information needed by the user.

**The next level of binding** comes when the search statement is parsed for use by a specific search system. The search system translates the query to its own meta language. For example, statistical systems determine the processing tokens of interest and the weights assigned to each processing token based upon frequency of occurrence from the search statement. Natural language systems determine the syntactical and discourse semantics using algorithms similar to those used in indexing. Concept systems map the search statement to the set of concepts used to index items.

**The final level of binding** comes as the search is applied to a specific database. This binding is based upon the statistics of the processing tokens in the database and the semantics used in the database. This is especially true in statistical and concept indexing systems. Some of the statistics used in weighting are based upon the current contents of the database. Some examples are Document Frequency and Total Frequency for a specific term. Frequently in a concept indexing system, the concepts that are used as the basis for indexing are determined by applying a statistical algorithm against a representative sample of the database versus being generic across all databases.

The length of search statements directly affects the ability of Information Retrieval Systems to find relevant items. The longer the search query, the easier it is for the system to find items. Profiles used as search statements for Selective Dissemination of Information systems are usually very long, typically 75 to 100 terms. In large systems used by research specialists and analysts, the typical ad hoc search statement is approximately 7 terms.



EExamples of Query Binding

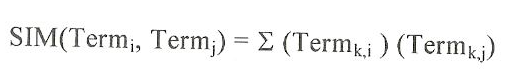
***Similarity Measures and Ranking***

Searching in general is concerned with calculating the similarity between a user's search statement and the items in the database. Although many of the older systems are unweighted, the newer classes of Information Retrieval Systems have logically stored weighted values for the indexes to an item. The similarity may be applied to the total item or constrained to logical passages in the item.

Once items are identified as possibly relevant to the user's query, it is best to present the most likely relevant items first. This process is called "ranking." Usually the output of the use of a similarity measure in the search process is a scalar number that represents how similar an item is to the query.

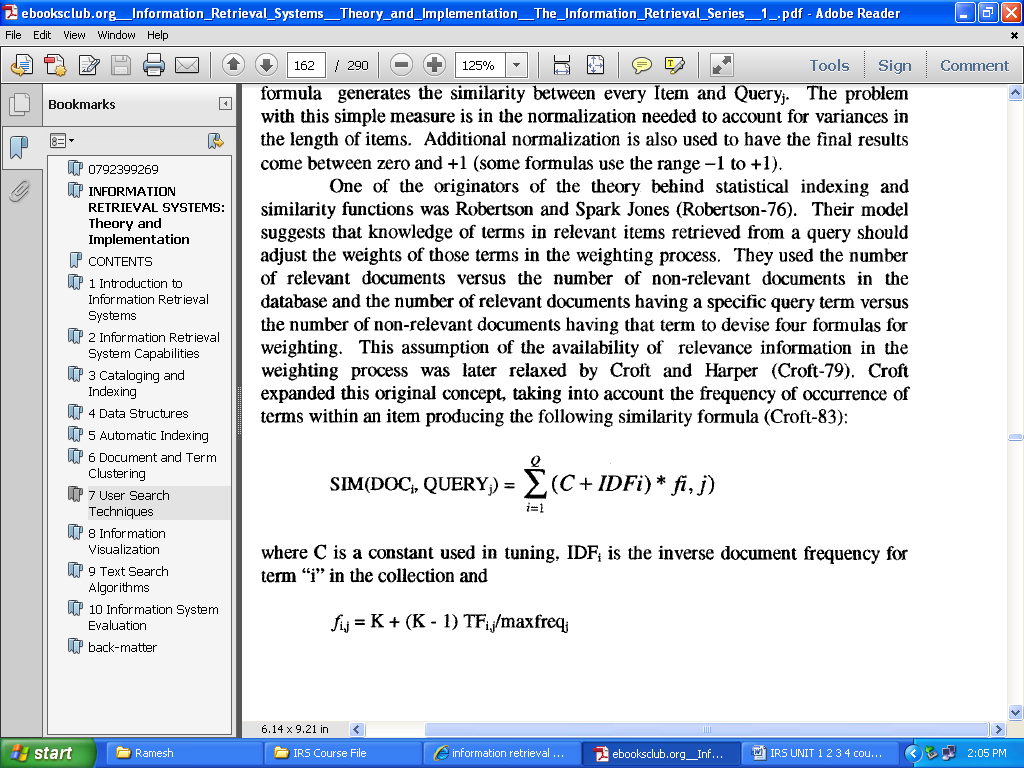
***Similarity Measures:***

A variety of different similarity measures can be used to calculate the similarity between the item and the search statement. A characteristic of a similarity formula is that the results of the formula increase as the items become more similar. The value is zero if the items are totally dissimilar. An example of a simple "sum of the products" similarity measure from the examples in clustering to determine the similarity between documents for clustering purposes is:



This formula uses the summation of the product of the various terms of two items when treating the index as a vector. If Itemj is replaced with Queryj then the same formula generates the similarity between every Item and Queryj. The problem with this simple measure is in the normalization needed to account for variances in the length of items. Additional normalization is also used to have the final results come between zero and +1 (some formulas use the range -1 to +1).

One of the originators of the theory behind statistical indexing and similarity functions was Robertson and Spark Jones (Robertson-76). Their model suggests that knowledge of terms in relevant items retrieved from a query should adjust the weights of those terms in the weighting process. They used the number of relevant documents versus the number of non-relevant documents in the database and the number of relevant documents having a specific query term versus the number of non-relevant documents having that term to devise four formulas for weighting. This assumption of the availability of relevance information in the weighting process was later relaxed by Croft and Harper (Croft-79). Croft expanded this original concept, taking into account the frequency of occurrence of terms within an item producing the following similarity formula (Croft-83):

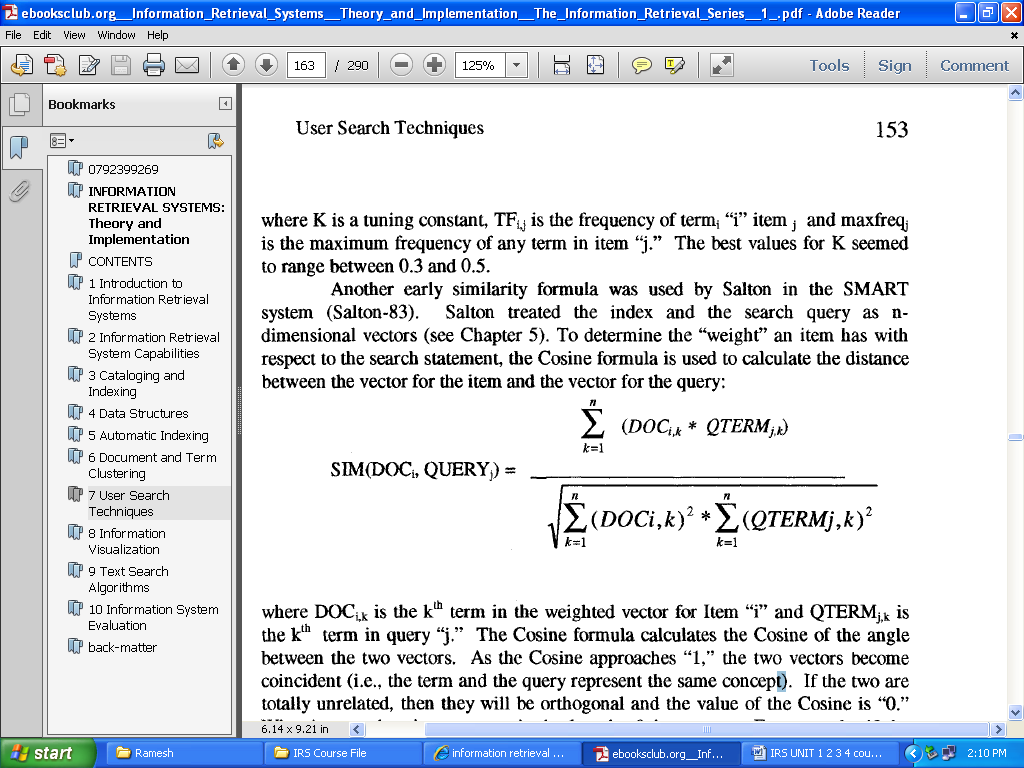


Where C is a constant used in tuning, IDFi is the inverse document frequency for term "i" in the collection and

Fi,j = K + (K - 1) TFij / maxfreqj

Where K is a tuning constant, TFi,j is the frequency of termi "i" item j and maxfreqi is the maximum frequency of any term in item "j." The best values for K seemed to range between 0.3 and 0.5.

Another early similarity formula was used by Salton in the SMART system (Salton-83). To determine the "weight" an item has with respect to the search statement, the Cosine formula is used to calculate the distance between the vector for the item and the vector for the query.



where DOC~k is the k m term in tile weighted vector for Item "i" and QTERMj,R is the k th term in query "j." The Cosine formula calculates the Cosine of the angle between the two vectors. As the Cosine approaches "1," the two vectors become coincident (i.e., the term and the query represent the same concep0. If the two are totally unrelated, then they will be orthogonal and the value of the Cosine is "0." What is not taken into account is the length of the vectors. For example, if the following vectors are in a three dimensional (three term) system:

Item = (4,8, 0)

Query 1 = (1, 2, 0)

Query 2 = (3, 6, 0)

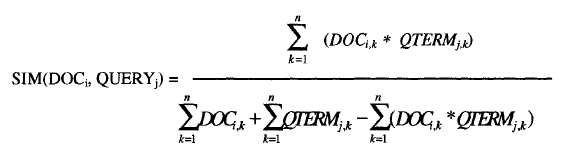
then tile Cosine value is identical for both queries even though Query 2 has significantly higher weights in the terms in common. To improve the formula, Salton and Buckley (Salton-88) changed the term factors in the query to:

QTERMi,k = (0.5 + (0.5 TFi./maxfreqk)) \* IDFi

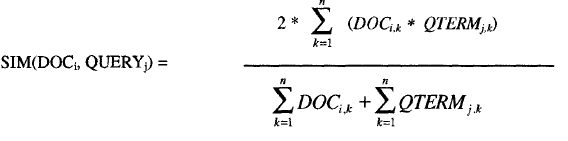
where TFi,k is the frequency of term "i" in query "k," maxfreqk is the maximum frequency of any term in query "k" and IDFi is the inverse document frequency for term "i" (see Chapter 5 for the formula). In the most recent evolution of the formula, the IDF factor has been dropped (Buckley-96).

Two other commonly used measures are the Jaccard and the Dice similarity measures. Both change the normalizing factor in the denominator to account for different characteristics of the data. The denominator in the Cosine formula is invariant to the number of terms in common and produces very small numbers when the vectors are large and the number of common elements is small. In the Jaccard similarity measure, the denominator becomes dependent upon the number of terms in common. As the common elements increase, the similarity value quickly decreases, but is always in the range -1 to +1.

The Jaccard formula is:

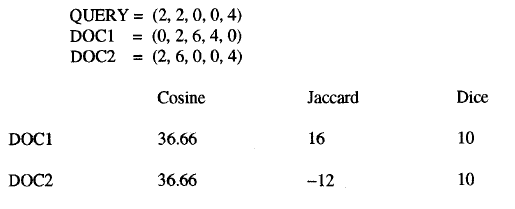


The Dice measure simplifies the denominator from the Jaccard measure and introduces a factor of 2 in tile numerator. The normalization in tile Dice formula is also invariant to the number of terms in common.

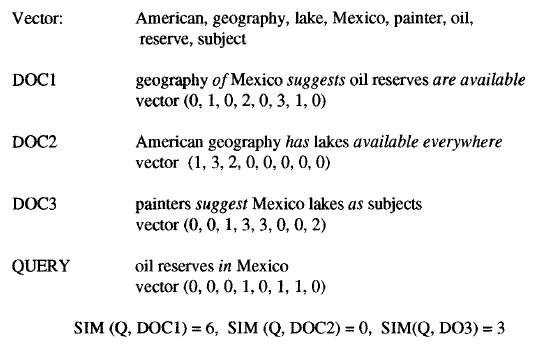


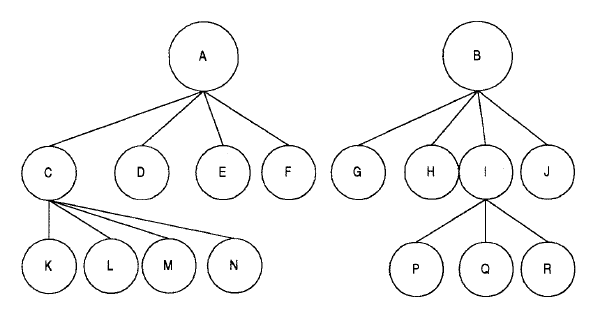
Use of a similarity algorithm returns tile complete data base as search results. Many of file items have a similarity close or equal to zero (or minimum value the similarity measure produces). For this reason, thresholds are usually associated with the search process. The threshold defines the items in the resultant

Hit file from the query. Thresholds are either a value that the similarity measure must equal or exceed or a number that limits the number of items in the Hit file. A default is always the case where the similarity is greater than zero. Figure below illustrates the threshold process. The simple "sum of the products" similarity formula is used to calculate similarity between the query and each document. If no threshold is specified, all three documents are considered hits. If a threshold of 4 is selected, then only DOC1 is returned. One special area of concern arises from search of clusters of terms that are stored in a hierarchical scheme



The items are stored in

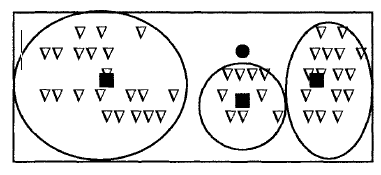




Item Cluster Hierarchy

In Figure below the filled circle represents the query and the filled boxes represent the centroids for the three clusters represented by the ovals. In this ease, the query may only be similar enough to the end two circles for additional analysis. But there are specific items in the right cluster that are much closer to the query than the cluster centroid and could satisfy the query. These items cannot be returned because when their centroid is eliminated they are no longer considered.

As part of investigating improved techniques to present Hits to users, Hearst and Pedersen from XEROX Palo Alto Research Center (PARC) are:





* **Ranking Algorithms:**

A by-product of use of similarity measures for selecting Hit items is a value that can be used in ranking the output. Ranking the output implies ordering the output from most likely items that satisfy the query to least likely items. This reduces the user overhead by allowing the user to display the most likely relevant items first. The original Boolean systems returned items ordered by date of entry into the system versus by likelihood of relevance to the user's search statement.

With the inclusion of statistical similarity techniques into commercial systems and the large number of hits that originate from searching diverse corpora, such as the Internet, ranking has become a common feature of modern systems. A summary of ranking algorithms from the research community is found in an article written by Belden and Croft.

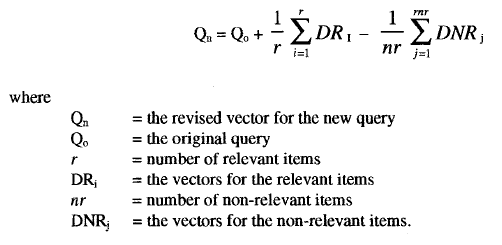
Fine grain ranking considers the physical location of query terms and related words using factors of proximity in addition to the other three factors in coarse grain evaluation, ff the related terms and query terms occur in close proximity (same sentence or paragraph) the item is judged more relevant. A factor is calculated that maximizes at adjacency and decreases as the physical separation increases. If the query terms are widely distributed throughout a long item, it is possible for the item to have a fine grain rank of zero even though it contains the query terms.

Although ranking creates a ranking score, most systems try to use other ways of indicating the rank value to the user as Hit lists are displayed. The scores have a tendency to be misleading and confusing to the user. The differences between the values may be very close or very large. It has been found to be better to indicate the general relevance of items than to be over specific.

**Relevance Feedback:**

Thesuari and semantic networks provide utility in generally expanding a user's search statement to include potential related search terms. But this still does not correlate to the vocabulary used by the authors that contributes to a particular database. There is also a significant risk that the thesaurus does not include the latest jargon being used, acronyms or proper nouns. In an interactive system, users can manually modify an inefficient query or have the system automatically expand the query via a thesaurus. The user can also use relevant items that have been found by the system (irrespective of their ranking) to improve future searches, which is the basis behind relevance feedback. Relevant items (or portions of relevant items) are used to reweight the existing query terms and possibly expand the user's search statement with new terms.

The relevance feedback concept was that the new query should be based on the old query modified to increase the weight of terms in relevant items and decrease the weight of terms that are in non-relevant items. This technique not only modified the terms in the original query but also allowed expansion of new terms from the relevant items. The formula used is:



The factors r and *nr* were later modified to be constants that account for the number of items along with the importance of that particular factor in the equation. Additionally a constant was added to Qo to allow adjustments to the importance of the weight assigned to the original query. This led to the revised version of the formula:

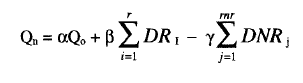
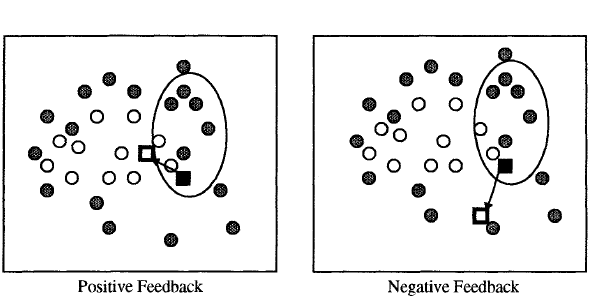


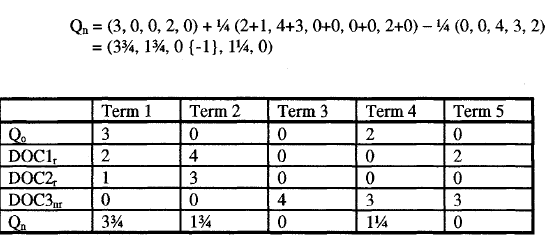
Figure below gives an example of the impacts of positive and negative feedback. The filled circles represent non-relevant items; the other circles represent relevant items. The oval represents the items that are returned from the query. The solid box is logically where the query is initially. The hollow box is the query modified by relevance feedback (positive only or negative only in the Figure).



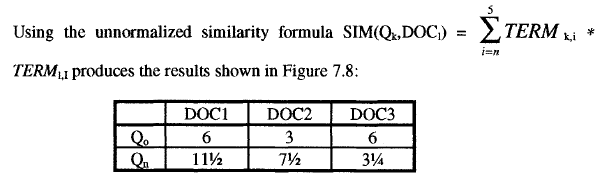


Positive feedback moves the query to retrieve items similar to the items retrieved and thus in the direction of more relevant items. Negative feedback moves the query away from the non-relevant items retrieved, but not necessarily closer to more relevant items.

Figure below shows how the formula is applied to three items (two relevant and one non-relevant). If we use the factors α = 1, β = 1/4 (½ times a constant ½), γ=1/4, in the foregoing formula we get the following revised query







Effect of Relevance Feedback

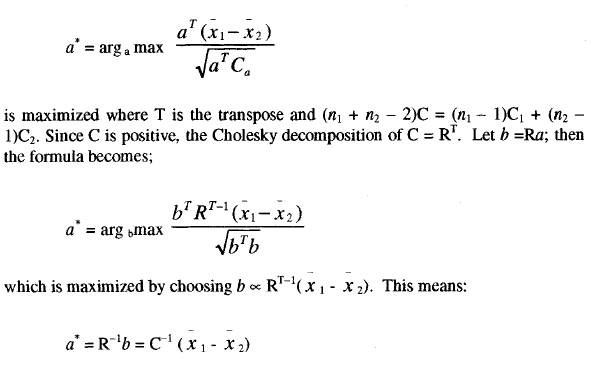
* **Selective Dissemination of Information Search:**

Selective Dissemination of Information, frequently called dissemination systems, are becoming more prevalent with the growth of the Internet. A dissemination system is sometimes labeled a "push" system while a search system is called a "pull" system. The differences are that in a search system the user proactively makes a decision that he needs information and directs the query to the information system to search. In a dissemination system, the user defines a profile (similar to a stored query) and as new information is added to the system it is automatically compared to the user's profile. If it is considered a match, it is asynchronously sent to tile user's "mail" file search statements (query and) is the introduction of a time parameter associated with a search statement.

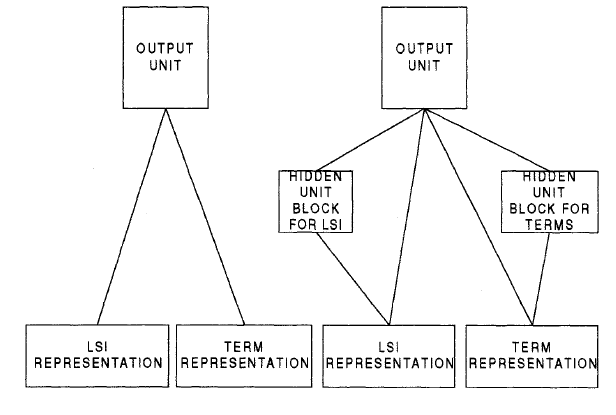
As long as the time is in the future, the search statement can be considered active and disseminating as items arrive. Once the time parameter is past, the user's need for tile information is no longer exists except upon demand between the two functions lie in the dynamic natprofiling process, the size and diversity of the search statements and number of simultaneous searches per item. In the search system, an existing database exists.

As such, corpora statistics exist on term frequency within and between terms. These can be used for weighting factors in the indexing process and tile similarit comparison (e.g., inverse document frequency algorithms). A dissemination system does not necessarily have a retrospective database associated with it. Thus its algorithms need to avoid dependency upon previous data or develop a technique to estimate terms for their formula. This class of system is also discussed as a binary classification system because there is no possibility for real time feedback from the user to assist in search statement refinement. The system makes a binary decision to reject or file tile item Once the reduced vector set has been identified, then learning algorithms can be used for the classification process.

Linear discriminate analysis, logistic regression and neural networks are three possible techniques that were compared by Schutze et al. Other possible techniques are classification trees, Bayesian networks, Bayesian classifiers, rules induction, nearest neighbor techniques, and least square methods. Linear discrimination analysis uses the covariance class for each document class to detect feature dependence. Assuming a sample of data from two groups with *n1* and *n2* members, mean vectors ˆx1 and ˆx2 and covariance matrices C1 and C2 respectively, the objective is to maximize the separation between the two groups. This can be achieved by maximizing the distance between the vector means, scaling to reflect the structure in the pooled covariance matrix. Thus choose such that:



A third technique is to use neural networks for the learning function. A neural network is a network of input and output cells (based upon neuron functions in the brain) originating with the work of McCuUoch and Pitts. Each input pattern is propagated forward through the network. When an error is detected it is propagated backward adjusting the cell parameters to reduce the error, thus achieving learning. This technique is very flexible and can accommodate a wide range of distributions. A major risk of neural networks is that they can overfit by learning the characteristics of the training data set and not be generalized enough for the normal input of items. In applying training to a neural network approach, a validation set of items is used in addition to the training items to ensure that overfitting has not occurred. As each iteration of parameter adjustment occurs on the training set, the validation set is retested. Whenever the errors on the validation set increase, it indicates that overfitting is occurring and establishes the number of iterations on training that improve the parameter values while not harming generalization. The linear and non-linear architectures for an implementation of neural nets is shown in Figure below:





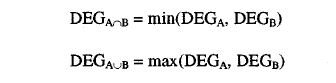
In the non-linear network, each of the hidden blocks consists of three hidden units. A hidden unit can be interpreted as feature detectors that estimate the probability of a feature being present in the input. Propagating this to the output unit can improve the overall estimation of relevance in the output unit. The networks show input of both terms and the LSI representation (reduced feature set).

In both architectures, all input units are directly connected to the output units. Relevance is computed by setting the activations of the input units to the document's representation and propagating the activation through the network to the output unit, then propagating the error back through the network using a gradient descent algorithm.

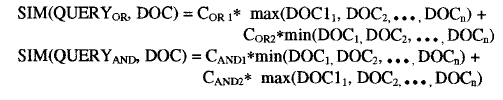
* **Weighted Searches of Boolean Systems:**

The two major approaches to generating queries are Boolean and natural language. Natural language queries are easily represented within statistical models and are usable by the similarity measures discussed. Issues arise when Boolean queries are associated with weighted index systems. Some of the issues are associated with how the logic (AND, OR, NOT) operators function with weighted values and how weights are associated with the query terms. If the operators are interpreted in their normal interpretation, thay act too restrictive or too general (i.e., AND and OR operators respectively). Salton, Fox and Wu showed that using the strict definition of the operators will suboptimize the retrieval expected by the user. Closely related to the strict definition problem is the lack of ranking that is missing from a pure Boolean process. Some of the early work addressing this problem recognized the fuzziness associated with mixing Boolean and weighted systems

To integrate the Boolean and weighted systems model, Fuzzy sets introduce the concept of degree of membership to a set (Zadeh-65). The degree of membership for AND and OR operations are defined as:

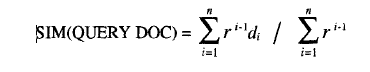


where A and B are terms in an item. DEG is the degree of membership. The Mixed Min and Max (MMM) model considers the similarity between query and document to be a linear combination of the minimum and maximum item weights. Fox proposed the following similarity formula:



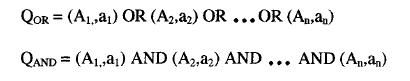
where COR1 and COR2 are weighting coefficients for the OR operation and CAND1 and CAND2 are the weighting coefficients for the AND operation. Lee and Fox found in their experiments that the best performance comes when CAreD1 is between 0.5 to 0.8 and Coal is greater than 0.2.

The MMM technique was expanded by Paice considering all item weights versus the maximum/minimum approach. The similarity measure is calculated as:

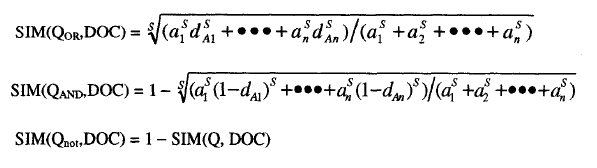


where the di’s are inspected in ascending order for AND queries and descending order for OR queries. The r terms are weighting coefficients. Lee and Fox showed that the best values for r are 1.0 for AND queries and 0.7 for OR queries (Lee-88). This technique requires more computation since the values need to be stored in ascending or descending order and thus must be sorted.

An alternative approach is using the P-norm model which allows terms within the query to have weights in addition to the terms in the items. Similar to the Cosine similarity technique, it considers the membership values (dA1 . . . . .dAn) to be coordinates in an "n" dimensional space. For an OR query, the origin (all values equal zero) is the worst possibility. For an AND query the ideal point is the unit vector where all the Di values equal 1. Thus the best ranked documents will have maximum distance from the origin in an OR query and minimal distance from the unit vector point. The generalized queries are:



The operators (AND and OR) will have a strictness value assigned that varies from 1 to infinity where infinity is the strict definition of the Boolean operator. The ai values are the query term weights. If we assign the strictness value to a parameter labeled "S" then the similarity formulas between queries and items are:



* **Searching the INTERNET and Hypertext:**

The Internet has multiple different mechanisms that are the basis for search of items. The primary techniques are associated with servers on the Internet that create indexes of items on the Internet and allow search of them.

The retrieved text is then used to create an index to the source items storing the Universal Resource locator (URL) to provide to the user to retrieve an item. All of the systems use some form of ranking algorithm to assist in display of the retrieved items. The algorithm is kept relatively simple using statistical information on the occurrence of words within the retrieved text.

Closely associated with the creation of the indexes is the technique for accessing nodes on the Internet to locate text to be indexed. This search process is also directly available to users via Intelligent Agents. Intelligent Agents provide the capability for a user to specify an information need which will be used by the Intelligent Agent as it independently moves between Internet sites locating information of interest. There are six key characteristics of intelligent agents.

* + 1. Autonomy - the search agent must be able to operate without interaction with a human agent. It must have control over its own internal states and make independent decisions. This implies a search capability to traverse information sites based upon pre-established criteria collecting potentially relevant information.
    2. Communications Ability - the agent must be able to communicate with the information sites as it traverses them. This implies a universally accepted language defining the external interfaces (e.g., Z39.50).
    3. Capacity for Cooperation - this concept suggests that intelligent agents need to cooperate to perform mutually beneficial tasks.
    4. Capacity for Reasoning - There are three types of reasoning scenarios:
    - Rule-based - where user has defined a set of conditions and actions to be taken
    - Knowledge-based - where the intelligent agents have stored previous conditions and actions taken which are used to deduce future actions
    - Artificial evolution based - where intelligent agents spawn new agents with higher logic capability to perform its objectives.

1. Adaptive Behavior - closely tied to 1 and 4, adaptive behavior permits the intelligent agent to assess its current state and make decisions oil the actions it should take
2. Trustworthiness - the user must trust that the intelligent agent will act on the user's behalf to locate information that the user has access to and is relevant to the user.