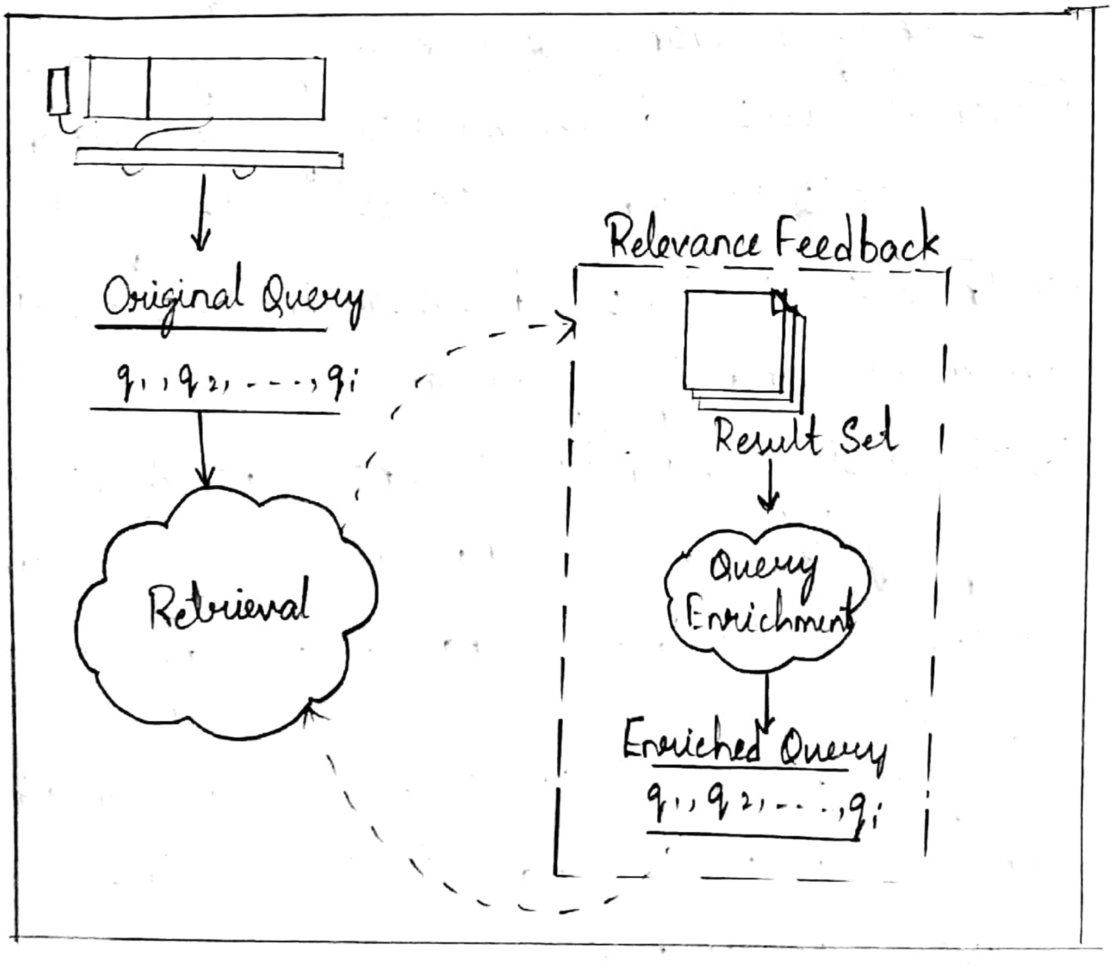
Relevance Feedback :

The basic premise of this model is to implement retrieval in multiple phases. The top documents found by an initial query are identified as relevant. These documents are then examined.

→ They may be deemed relevant either by manual intervention (consumption) or by an assumption that the top 'n' documents are relevant.

→ The ~~basic~~ user defines multiple phases.



→ The user defines a query in each pass based on result of previous queries.

→ Typically the user indicates which of the documents presented in response to an initial query or relevant and new terms are added to query based on this selection.

Relevance Feedback in Vector Space Modelling :

→ This approach used the vector space model to rank documents.

→ The query is represented by vector queue.

→ Each document is represented by vector D_i and a measure of relevance between a query and document vector is computed as similarity co-efficient.

$SC(Q, D_i)$ where SC is similarity Co-efficient.

→ There SC is computed as an inner product of the document and query vector.

→ The basic assumption is that the user has issued a query 'Q' and retrieved a set of documents.

→ The user is then asked whether the documents are relevant.

→ After the user response the set 'R' contains n_1 relevant documents and set 'S' contains n_2 non-relevant documents.

→ The new query Q' and old query Q using equation.

$$Q' = Q + \frac{1}{n_1} \sum_{i=1}^{n} R_i - \frac{1}{n_2} \sum_{i=1}^{n_2} S_i$$

→ The process can be repeated with 'i' number of repititions such that Q_{i+1} derived from Q_i for as many iterations as desired.

→ In addition to use the values in n_1 and n_2. It is possible to use arbitary weights.

$$Q' = \alpha Q + \beta \sum_{i=1}^{n_1} \frac{R_i}{n_1} - \gamma \sum_{i=1}^{n_2} \frac{S_i}{n_2}$$

→ The weights α, β, γ are Rocchio weights and the optimal values experimentally obtained but it is considered today to drop the use of non-relevant documents. So assign the value of γ as '0' and only use relevant document.

→ This basic theme was used by Rocchio. where

$$Q' = \alpha Q + \beta \sum_{i=1}^{n_1} R_i - S_1.$$

→ Only top ranked non-relevant documents are used instead of sum of all relevant documents.

Relevance Feedback in Probabilistic Model:

→ This model is well suited for relevance feedback it is necessary to know how many relevance documents exist for query to compute the term weight.

→ Typically the native probabilistic model requires some training data for which relevance info is known.

→ Once the term weights are computed they are applied to another collection. Relevance feedback doesnot required training data viewed as simply a utility instead of retrieval strategy.

→ Probabilistic relevance pluggins to any existing retrieval strategy.

→ The initial query is executed using an orbital retrieval strategy. and then the relevant information obtained during the feedback stage is incorporated for example, the basic weight used in probabilisti retrieval strategy is

$$w_i = \log \left[\frac{\frac{r_i}{R - r_i}}{\frac{n_i - r_i}{(N - n_i) - (R - r_i)}} \right]$$

where,

w_i = weight of term i in a particular query

R = number of documents that are relevant to the query.

N = number of documents in collection

r_i = number of relevant documents that contain term i.

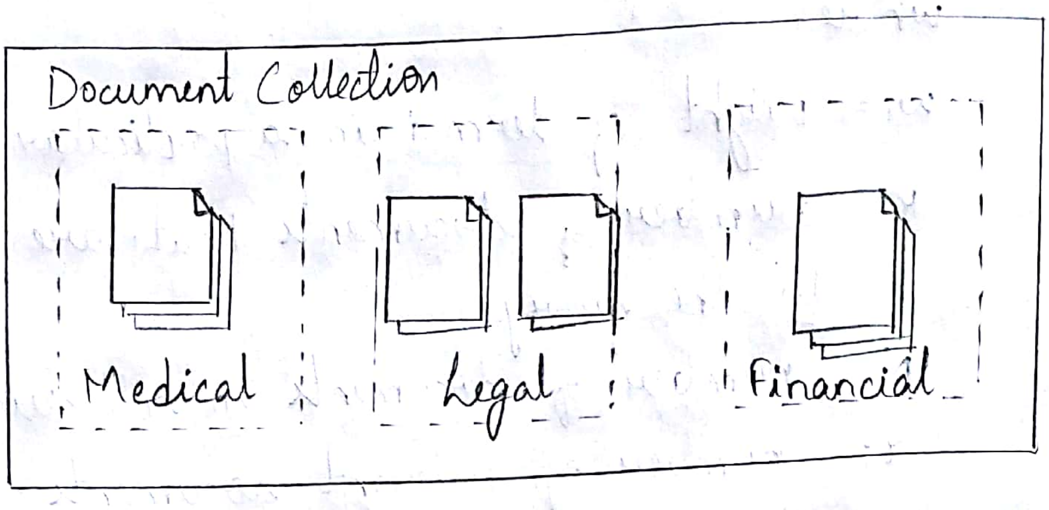
n_i = number of documents that contain term i.

→ Initially R, r cannot be known at the time of initial query unless training data with relevance information is available.

Clustering :

Document clustering attempts to group documents by content to reduce the each space required to respond the query.

→ For example, a document collection that contains both medical and legal documents might be clustered such that all medical documents are placed into one cluster and all legal documents are assigned to legal cluster.



→ A query over a legal material might be directed either automatically or manually to legal document cluster.

→ Several clustering algorithms have been proposed In many cases the evaluation of clustering algorithms have been challenging because it is difficult pointing a query at document cluster.

→ Many algorithms began with a matrix that contains the similarity of each documents with every other document

Result Set Clustering :

Clustering was used as utility to assist relevance feedback. Recently, web search results were clustered based on significant phrases in result set. First documents in a result set are parsed and two term phrases are identified. The characteristics are then used as input to model built by various learning algorithms. Once the most significant phrases are identified they are used to build clusters. A cluster is initially ~~identified~~ defined as set of documents.

Hierarchical Agglomerative Clustering:

First N×N document similarity matrix is formed. Each document is placed into its own cluster. The following steps are repeated until one cluster exists.

1. The two clusters that have the highest similarity are found.
2. These two clusters are combined, and similarity between newly formed cluster and remaining clusters recomputed.

(i) Single Link Clustering:

The similarity between two clusters are computed as maximum similarity of any two documents in two clusters.

(ii) Complete Linkage:

Inter cluster similarity is computed as minimum of similarity between any documents in two clusters such that one document is from each cluster.

(iii) Group Average:

A node is considered for each cluster its average similarity to all nodes in that cluster is computed. It is placed in cluster as long as its average similarity is higher than its average similarity for any other cluster.

(iv) Ward's Method:

Clusters are joined so that their merger minimizes the increase in the sum of distances from each individual document to the centroid of cluster containing it.

$$c_j = \frac{\sum_{i=1}^{n} t_{ij}}{n}$$

Clustering Without a Precomputed Matrix:

(i) One-Pass Clustering:

The document is assumed to

be in a cluster of size one. The similarity is computed as distance between the new document and the centroid of existing clusters.

Rocchio Clustering:

In this, all documents are scanned and defined as either clustered or loose. An unclustered document is tested as potential center of cluster by examining the density of the document.

K-Means:

It is partitioning algorithm that moves k centroids until a termination condition is met

BuckShot Clustering:

It is designed so that it runs in $O(kn)$ time where k is number of clusters that are generated and n is number of documents. For applications, where number of desired clusters is small the clustering time is close to $O(n)$ which is a clear improvement over $O(n^2)$.

Non - negative Matrix Factorisation :

It provides a latent semantic space where it represents the topic of each cluster. Documents are represented as sum of each axis and are assigned to cluster.

N-grams

Term-based search techniques typically use an inverted index or a scan of the text. Additionally, queries that are based on exact matches within terms in a document perform poorly against corrupted documents. This occurs regard-less of the source of the errors - either OCR (optical character recognition) errors or those due to misspelling. To provide resilience to noise, n-grams were proposed. The premise is to decompose terms into word fragments of size n, then design matching algorithms that use these fragments to determine whether or not a match exists.

N-grams have also been used for detection and correction of spelling errors and text compression. A survey of automatic correction techniques is found in Kukich. Additionally, n-grams were used to determine the authorship of documents. Traditional information retrieval algorithms based on n-grams are described in D'Amore and Mah, 1985, Damashek, 1995, Pearce and Nicholas, 1993, Teuful, 1988, Cavnar and Vadya, 1993.

Uses of N-grams

a) This mechanism was widely used for identifying and rectifying the errors in spellings.

b) It was used for performing the text compressions.

c) It was used for obtaining the information regarding authorization of the documents.

D' Amore and Mah method

The method presented by D' Amore and Mah focussed on eliminating the difficulties involved in establishing mathematical models for terms based retrievals. These mathematical models for terms based retrievals. These mathematical models can be used for computing the amount of noise involved in indexing and also to compute the similarity between the documents.

From all these computations, the ultimate weight of an n-gram 'i' present in a document (j) can be given as,

$$\boxed{w_{ij} = \frac{f_{ij} - e_{ij}}{\sigma_{ij}}}$$

where,

f_{ij} is the frequency of n-gram (i) in document (j)

e_{ij} is the expected number of occurances of an n-gram 'i' in document 'j'.

σ_{ij} is the standard derivation.

Damashek Method:

The method proposed by Damashek can be viewed as the extension of D'Amore and Mah's method that employs five-gram instead of n-gram based measure of relevance for information retrieval. the difference lies in the way relevance is computed. This method eradicates the noise by using the centroid vectors rather than stop words (or) stemming algorithms for regulating the no. of occurances of n-grams. Also, the similarity coefficient is computed as the cosine measure using the following formula.

$$\text{simcoeff}(Q,D) = \frac{\sum_{j=1}^{t}(w_{qj}-\mu_Q)(w_{dj}-\mu_D)}{\sqrt{\sum_{j=1}^{t}(w_{qj}-\mu_q \sum_{j=1}^{t}(w_{dj}-\mu_D)^2}}$$

where

μ_q and μ_d = centroid vectors used to differentiate between languages of query and document

w_{qj} and w_{dj} = weight of the terms in query and document vectors.

Regression Analysis

Another approach to estimating the probability of relevance is to develop variables that describe the characteristics of a match to a relevant document. Regression analysis is then used to identify the exact parameters that match the training data. For example, if trying to determine an equation that predicts a person's life expectancy given their age:

Name	Age	Value
Ram	60	6
Raj	65	5

A simple least squares polynomial regression could be implemented that would identify the correct values of α and β to predict life expectancy (LE) based on age (A):

$$LE = \alpha A + \beta$$

The above tabular data can be extended by considering the information regarding the retirement of an employee based upon age (or) experience. The extended table can be given as follows.

Then a line (or) a curve can be fitted for the above tabular data. whenever a new employee tries to retrieve data regarding their retirement, the corresponding point of age (or) experience can be analysed from the curve (or) the line. This comes under logistic type of regression which is related to document retrieval. Using this logistic regression, dichotomus variables can be computed. These are the type of variables that contain smaller set of values within them. 1

These are of 6 types.

1. Mean of total number of terms matching in query.

2. Square root of overall terms involved in query.

3. Mean of total number of terms matching in document.

4. Square root of overall terms ~~matchi~~ involved in document.

5. Average idf of terms matching.

6. Total number of terms matching in the query.

Every variable needs to be independent of others in this approach. Using these variables, logistic regression can be performed in two stages. In the first ~~age~~ stage, composite clues comprising the independent variables are computed.

Stage 1: Composite clue are computed. For each composite clue logistic regression is carried out.

$$\log O(Rel/CC_1) = a_0 + a_1 X_1 + a_2 X_2 + a_3 X_3$$

$$\log O(Rel/CC_2) = b_0 + b_1 X_4 + b_2 X_5 + b_3 X_6$$

where Rel = Relevance

CC_1 = Composite clue 1

CC_2 = Composite clue 2

a_0, a_1, a_2, a_3 = coefficients used for computing relevance for composite clue CC_1.

b_0, b_1, b_2, b_3 = coefficients used for computing relevance for composite clue CC_2.

Stage 2: At this stage of logistic regression, the errors from composite clues will be identified and rectified. The no. of errors identified is directly proportional to the no. of composite clues. If there are 'N' composite clues, then the logistic regression is estimated as

$$\log O(Rel/CC_1, CC_2, \cdots CC_N) = a_0, a_1 Z + a_2 N.$$

where

Rel = Relevance

CC_1 = Composite clue 1

CC_2 = composite clue 2

CC_N = composite clue N

a_0, a_1, a_2 = Coefficients to estimate relevance

Z = sum of composite clues

$= \sum_{i=1}^{N} \log O(Rel / CC_i)$

The output obtained from the first stage can be applied to next stage and thus any no.of stages can be involved.

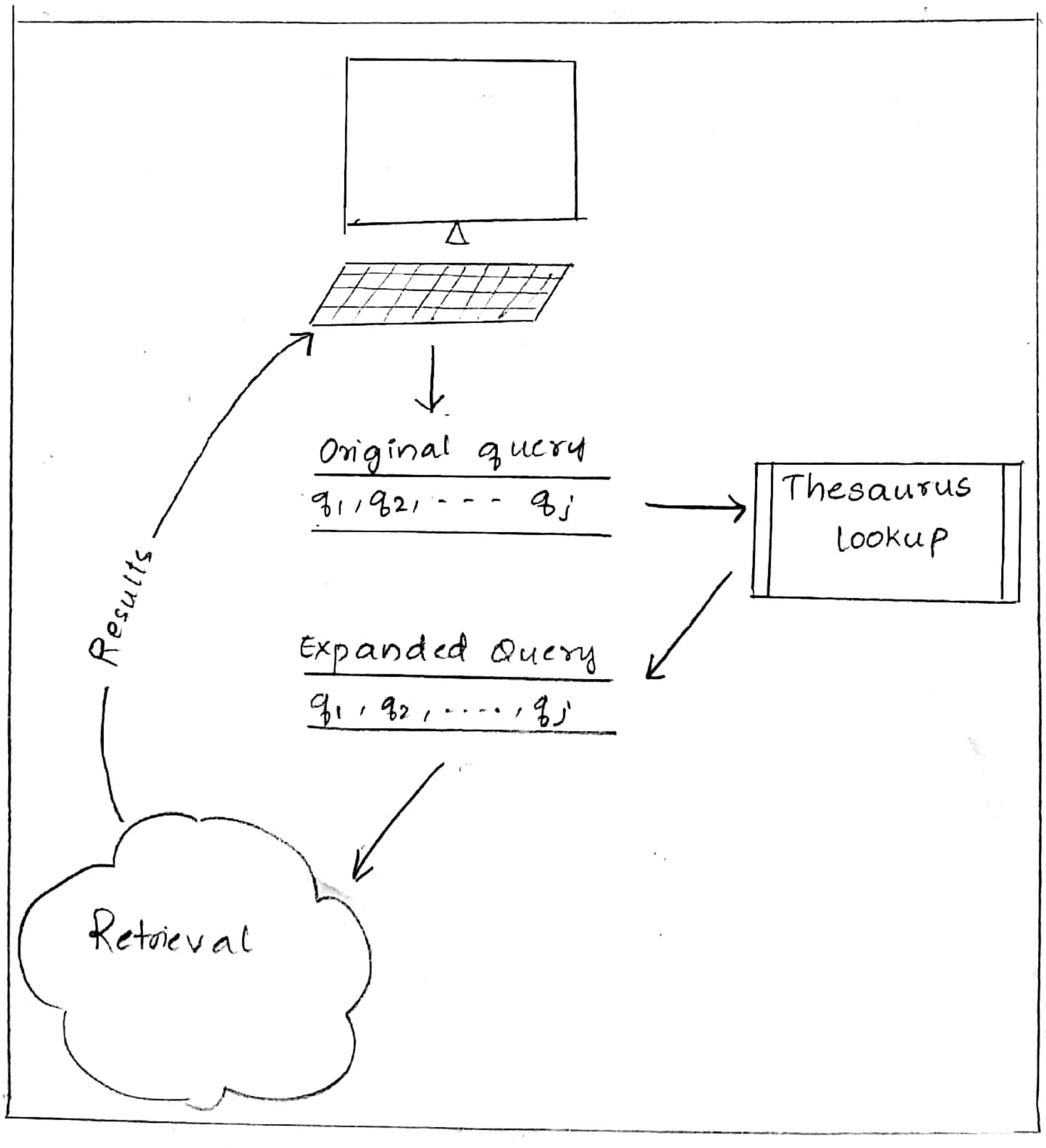
~~Adva~~

Advantages:

1. Best clues can be recognised by performed experiments.
2. It enables to rectify the errors found in the initial logistic regression.
3. It allows to avoid the basic independence assumptions.

Thesauri :

One of the most intuitive ideas for enhancing effectiveness of an information retrieval system is to include the use of a thesaurus. Almost from the drawn of the first information retrieval systems in the early 1960's, researchers focuses on incorporating a thesaurus to improve precision and recall. The process of using a thesaurus to expand a query is illustrated in Fig.



Scanned by CamScanner

Automatically Constructed Thesauri :

Term Co-occurrence :

To form a thesaurus for a given term t, related terms for t are all those terms u such that SC (t,u) is above a given threshold. Note. this is an $O(t^2)$ process, so it is often common to limit the terms for which a related term list is built.

Consider the following example :

D_1 : "a dog will bark at a cat in a tree"

D_2 : "ants eat the bark of a tree"

To compute the similarity of term i with term j, a vector of size N, where N is number of documents, is obtained for each term. A dot product similarity between "bark" and "tree" is computed as :

$$SC(\text{bark}, \text{tree}) = \langle 1 \; 1 \rangle \cdot \langle 1 \; 1 \rangle = 2$$

The matrix is symmetric as $SC(t_1, t_2)$ is equivalent to $SC(t_2, t_1)$. Consider "tree" and "bark" in our example, these terms co-occur twice in two documents.