12-2011

Consolidated study on query expansion

Abhishek Biruduraju
University of Nevada, Las Vegas

CONSOLIDATED STUDY ON QUERY EXPANSION

by

Abhishek Biruduraju

Bachelor of Technology in Computer Science and Engineering
Jawaharlal Nehru Technological University, India
May 2008

A thesis submitted in partial fulfillment of the requirements for the

Masters of Science Degree in Computer Science
Department of Computer Science
Howard R. Hughes College of Engineering
Graduate College

University of Nevada, Las Vegas
December 2011

ABSTRACT

Consolidated study on query expansion

by

Abhishek Biruduraju

Dr. Kazem Taghva, Examination Committee Chair

Professor of computer science

University of Nevada, Las Vegas

A typical day of million web users all over the world starts with a simple query. The quest for information on a particular topic drives them to search for it, and in the pursuit of their info the terms they supply for queries varies from person to person depending on the knowledge they have. With a vast collection of documents available on the web universe it is the onus of the retrieval system to return only those documents that are relevant and satisfy the user's search requirements. The document mismatch problem is resolved by appending extra query terms to the original query which improves the retrieval performance. The addition of terms tends to minimize the bridging-gap between the documents and queries.

In this thesis, a brief study is done on the reformulation of queries, along with methods of calculating the relevancy of candidate terms for query expansion by using several ranking algorithms, term weighting algorithms and feedback processes involving evaluations. Comparisons of various methods based on their efficiencies are also discussed. On the whole a consolidated report of query expansion in general is given.

ACKNOWLEDGEMENTS

I would like to take this opportunity to thank Dr. Kazem Taghva, my research advisor for his constant guidance and support throughout the entire duration of the thesis work. His unwavering encouragement and effective advice kept me going.

I am also grateful to the faculty, librarians, and all the other staff at University of Nevada, Las Vegas.

I would also like to thank my friends and students at this university for the motivation they provided during the tough times and even for the fun and joyous environment that boosted me to learn and grow.

Lastly and most importantly, I would like to thank my family for the love and care showered on me and I would like to thank my mom for the sacrifices she made in trying to make me pursue my dreams. To them, I dedicate this thesis.

TABLE OF CONTENTS

List of Tables

Chapter 1: introduction and overview

Users all over the world are always in a quest to find out something relevant from the vast collections of data spread throughout the web universe. Information Retrieval (IR) is one such area of research which helps the user in finding documents that satisfies their needs. The IR systems are based on models of retrieval process. These models represent the way the documents are represented and compared against information needs in order to estimate the relevancy of the obtained document. The users try to extract information by providing queries in the form of terms to the system and these queries are intrinsically ambiguous to it, the inadequacy of the terms can cause the system to return likely high chances of deviated or irrelevant topics unless the initial terms are supplemented with additional terms that improves the retrieval performance as measured by its effectiveness.

This method of adding extra terms to the query has led to the instantiation of the concept of query expansion which is an effective method of retrieval. The search thus done can be staged in 2 ways

- Initial query formulation – the user prepares the search strategy
- Query reformulation – the user tries to adjust the initial query manually or with the assistance of the system or the system itself automatically adjusts the query for improved possible outcomes

1.1 Query expansion intro

Query expansion can take place in either of the two stages. Thus the query expansion can be manual or automatic or interactive. For any of these expansions to work there has to be a source for the terms and these sources are again classified into two types [9]

- Search results – documents retrieved in the first or an earlier iterations of search which have been deemed relevant become the source for the expansion
- Knowledge structures – as these are independent of the search, they can be either dependent or independent of the collection. The dependent knowledge structures are general algorithmic processes for word modification (suffix stripper, string similarity etc.) , term clusters , automatic constructed thesauri, and the independent knowledge structures are domain specific thesauri or dictionaries/lexicons.

From the above mentioned sources, the other important aspect of query expansion is the method of selecting terms to be added. There are several ranking algorithms discussed for it [4]. Testing these approaches can be done on search behaviors and experiments are conducted on query modification using retrieval techniques which can be either Boolean or weighted term. These techniques try to answer the doubts that arise like what are the best terms? , how do we rank them? , how do the user select them? Etc.

As the documents are a vast collection of terms, we are interested in selecting those indexed terms which act as good discriminators from relevant and non relevant documents. In manual expansion the users at their discretion select terms by the knowledge they possess with the help of searching aids like thesauri and try to choose terms relevant to the clusters. In automatic query expansion (AQE), the retrieval of terms can be based on the weight or associated queries or several methods which have been proposed by several authors based on their effectiveness. In interactive query expansion (IQE), both the user and system share the responsibility of selection [9].

Current techniques for query expansion use values for key parameters, determined by test collections. They show that these parameters may not be generally applicable, and that the assumption that the same parameter settings can be used for all queries is invalid. Using detailed experiments, demonstrated that new methods for choosing parameters must be found. In conventional approaches to query expansion, the additional terms are selected from highly ranked documents returned from an initial retrieval run. There are also methods of obtaining expansion terms, based on past user queries from query logs that are associated with documents in the collection. The most effective query expansion methods rely on retrieval and processing of feedback documents. The first retrieval conducted should return useful documents so that the query reformulation can be done for extracting useful items for the subsequent searches.

We study the concept of relevance feedback [5] to improve the effectiveness as it has some advantages like breaking down the search strategy into smaller search steps so that the gap of similarity between query and the documents is reduced as it emphasizes and de-emphasizes terms depending on the methods employed. We study vector and probabilistic methods for this. Then we evaluate the effectiveness of it using recall and precision. The catch here is that originally retrieved relevant item may be retrieved again with a better rank each time search progresses and this may not reflect the users' relevancy, so it must be judged by the ability to retrieve new unseen terms. There are six relevance feedback methods of interest here, each tested against different collections and their measuring parameters compared.

As all the documents and queries are indexed based on the terms, it is important to determine the importance of words which can be achieved based on the assignment of weights. Words extracted are used for content identification and we determine various possibilities for representations like related terms, phrases, thesauri or knowledge bases. Sometimes term dependencies are involved which are effectively valid only locally from documents from which original words were extracted. Term weighting systems are preferred that produce both high recall by retrieving documents that are relevant and also high precision by rejecting items that deviate the user from the intended topic. Some weighting factors are considered for enhancing both the measures like term frequency, inverse document frequency and the normalization factors. Term discrimination lies in the ability to distinguish and this means that best terms should have high frequency but low overall collection frequencies. In term weighting systems, both document and queries are represented by vectors of weighted terms. There are a few term-weighting components which are used to generate few formulas which make up the type of system. Experiments are then conducted using these systems against a collection set and we derive some conclusions based on their effectiveness.

Another methodology which is discussed is the use of real users with their real requests in an operational environment to study query expansion dynamically. The most important characteristic of it is the selection of terms using the constraints imposed by the user. The order of terms is such that the useful terms are at the top of the list. So apart from weighting the terms, ranking is also important and a few ranking algorithms like F4, porters, EMIM, ZOOM and WPQ are studied. Again the effectiveness is measured based on precision and recall. The ranks of the terms were added and the sum was used for comparisons. It is based on the notion that the sum of the terms would indicate the relative importance that each algorithm gives to the user preferences.

After studying the methodologies we study these query expansions by various types of expansions provided by several authors and discuss their results.

Chapter 2: IR MODELS

We know that the goal of an information retrieval system is to provide the users with the documents that satisfy the needs of the user. Users have to formulate their information need in a form that can be understood by the retrieval mechanism and the contents of large document collections need to be described in a form that allows the retrieval mechanism to identify the potentially relevant documents quickly. There are two major models which have been developed to retrieve information; they are Boolean models and statistical models.

2.1 The Boolean model

This was the first classical model which was adopted on most of the earlier systems and even today a lot of commercial systems use this model which makes use of the concepts of Boolean logic and set theories.

The documents and the queries are a collection of terms and each term from the document is indexed. The presence and absence of a term in a document is represented by 1 and 0 respectively. For the term matching of document and query we maintain an inverted index of the terms i.e. for each term we must store a list of documents that contain the term. The terms are tokenized using linguistic models for those terms which can be stemmed down. The sequence of terms can be identified as < term, document ID> which can be sorted too. We can also have another identifier like frequency.

Each query term specifies a set of documents containing the term and the Boolean operations performed on them are

- AND – the intersection of two sets
- OR – the union of two sets
- NOT – the set inverse or the set difference

A simple algorithm for AND would be as follows: For each query term t, note f_t (frequency) and sort the terms by increasing it. Initialize the candidate set with the address of the inverted list of the term with the smallest f_t. Then for the remaining terms, look for the terms in the documents from the candidate which does not contain the term and eliminate them. For queries which are of the form of conjunctions and disjunctions, treat each of the disjunction as a single term and merge the inverted lists for each OR-ed terms.

The strengths of this model are

- Easy to implement and computationally efficient.
- It enables users to express structural and conceptual constraints to describe important linguistic features
- The Boolean approach possesses a great expressive power and clarity. Boolean retrieval is very effective if a query requires an exhaustive and unambiguous selection.
- The Boolean method offers a multitude of techniques to broaden or narrow a query.

- The Boolean approach can be especially effective in the later stages of the search process, because of the clarity and exactness with which relationships between concepts can be represented.

The limitations of this model are

- Users find it difficult to construct effective Boolean queries for several reasons. Users are using the natural language terms AND, OR, NOT that have a different meaning when used in a query. Thus, users will make errors when they form a Boolean query.
- Only documents that satisfy a query exactly are retrieved. The AND operator does not distinguish between the case when none of the concepts are satisfied and the case where all except one are satisfied. Hence, no or very few documents are retrieved when more than three and four criteria are combined with the Boolean operator AND (referred to as the Null Output problem). On the other hand, the OR operator does not reflect how many concepts have been satisfied. Hence, often too many documents are retrieved (the Output Overload problem).
- It is difficult to control the number of retrieved documents. Users are often faced with the null-output or the information overload problem and they are at loss of how to modify the query to retrieve the reasonable number documents.
- The traditional Boolean approach does not provide a relevance ranking of the retrieved documents.

- It does not represent the degree of uncertainty or error due to the vocabulary problem.

2.2 The Statistical models

The vector space and probabilistic models are the two major examples of the statistical retrieval approach. Both models use statistical information in the form of term frequencies to determine the relevance of documents with respect to a query. Although they differ in the way they use the term frequencies, both produce as their output a list of documents ranked by their estimated relevance. The statistical retrieval models address some of the problems of Boolean retrieval methods, but they have disadvantages of their own too.

2.2.1 Vector space model

We know that the similarity function of a Boolean model is Boolean and hence we get "exact-matching" documents where as we have a different similarity function for the vector space model where documents and queries are represented in the form of vectors. The vector space model procedure can be divided into three stages.

1. The first stage is the document indexing where content terms are extracted from the document text. This indexing can be based on term frequency, where terms that have both high and low frequency within a document are considered to be function words. Non linguistic methods for indexing have also been implemented. Probabilistic indexing is based on the assumption

that there is some statistical difference in the distribution of content bearing words, and function words.

2. The second stage is the weighting of the indexed terms to enhance retrieval of document relevant to the user. The term weighting for the vector space model has entirely been based on single term statistics. There are three main factors for term weighting: term frequency factor, collection frequency factor and length normalization factor. These three factor are multiplied together to make the resulting term weight.
3. The third stage ranks the document with respect to the query according to a similarity measure. The similarity in vector space models is determined by using associative coefficients based on the inner product of the document vector and query vector, where word overlap indicates similarity. The inner product is usually normalized. The most popular similarity measure is the cosine coefficient, which measures the angle between the document vector and the query vector.

If D and Q are vectors of the document and query respectively,

$$d_t = (w_{t,1}, w_{t,2} .. w_{t,t})$$

$$\mathrm{Sim}(D, Q) = \sum_{i=1}^{t} d_i \cdot q_i = \frac{\sum w_{t,f} \times w_{q,f}}{\sum {w_{t,f}}^2 \cdot \sum {w_{q,f}}^2}$$

2.2.2 Probabilistic model

T he probabilistic retrieval model is based on the Probability Ranking Principle, which states that an information retrieval system is supposed to rank the documents based on their probability of relevance to the query. The principle takes into account that there is uncertainty in the representation of the information need and the documents. There can be a variety of sources of evidence that are used by the probabilistic retrieval methods, and the most common one is the statistical distribution of the terms in both the relevant and non-relevant documents. These probabilities are used to rank documents.

The similarity function is defined based on the relevancy of documents given by

$$\mathrm{sim}(d, q) = \frac{P(R|d)}{P(\bar{R}|d)}$$

The binary independence model is a probabilistic information retrieval technique. "Independence" signifies that terms in the document are considered independently from each other and no association between terms is modeled. The probability P(R|d,q) that a document is relevant derives from the probability of relevance of the terms vector of that document P(R|x,q). By using the Bayes rule we get

$$P(R|x, q) = \frac{P(x|R,q) \times P(R|q)}{P(x|q)}$$

where P(x|R=1,q) and P(x|R=0,q) are the probabilities of retrieving a relevant or non-relevant document, respectively. If so, then that document's representation is x. The exact probabilities cannot be known beforehand, so estimates from statistics about the collection of documents must be used. P(R=1|q) and P(R=0|q) indicate the previous probability of retrieving a relevant or non-relevant document respectively for a query q. If, for instance, we knew the percentage of relevant documents in the collection, then we could use it to estimate these probabilities. Since a document is either relevant or non-relevant to a query we have that:

$P(R = 1 \mid x,q) + P(R = 0 \mid x,q) = 1$

The statistical approaches have the following strengths

- They provide users with a relevance ranking of the retrieved documents. Hence, they enable users to control the output by setting a relevance threshold or by specifying a certain number of documents to display.
- Queries can be easier to formulate because users do not have to learn a query language and can use natural language.
- The uncertainty inherent in the choice of query concepts can be represented

The statistical approaches have the following limitations

- They have a limited expressive power. For example, the NOT operation cannot be represented because only positive weights are used.
- The statistical approach lacks the structure to express important linguistic features such as phrases. Proximity constraints are also difficult to express.
- The computation of the relevance scores can be computationally expensive.
- A ranked linear list provides users with a limited view of the information space and it does not directly suggest how to modify a query when necessary.
- The queries have to contain a large number of words to improve the retrieval performance. As is the case for the Boolean approach, users are faced with

the problem of having to choose the appropriate words that are also used in the relevant documents.

Chapter 3: Query Feedback

The information stored over the web is so vast that in most of the collections, the same concept may be represented in different words. This can be an issue which can impact the retrieval effectiveness of an IR system. The effectiveness is measured based on two factors which are known as recall and precision.

Before we delve into further topics, let’s define them both.

Precision: this performance measure is the fraction of documents retrieved that are relevant to the user’s information needs.

$$\text{precision} = \frac{\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}}{\{\text{retrieved documents}\}}$$

This can also be evaluated at a given cut-off value say n, considering only the top most documents returned by the system which we call as precison@n or P@n.

Recall: this performance measure, is the fraction of the documents that are relevant to the query that are retrieved successfully.

$$\text{recall} = \frac{\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}}{\{\text{relevant documents}\}}$$

This can be seen as the probability that a relevant document is retrieved by the system.

Precision and recall values can be plotted to give a precision-recall curve.

Hence the problems which impact the user's needs through these measures are to be tackled by refining the queries. These refinements can be manual or automatic.

The methods for tackling this problem can be through two major cases: global methods and local methods. Global methods are techniques for expanding or reformulating query terms independent of the query and the results returned from it, so that refinements in the query will cause the new query to match other semantically similar terms. Global methods include:

- Query expansion/reformulation with a thesaurus
- Query expansion via automatic thesaurus generation
- Spelling correction

Local methods make adjustments to a query in such a way that the queries relative to the documents that initially appear to match the query are obtained. The basic methods here are

- Relevance feedback
- Pseudo relevance feedback
- Indirect relevance feedback

We need to consider the following issues: relevance feedback; initial query terms and query expansion terms; relevance judgments, e.g., document representation to base the relevance judgments, methods for relevance assessments, sample size of documents for estimating weights, ranking algorithms and term selection for query expansion.

3.1 Relevance feedback

We have seen that the original query formulation process is not transparent to most of the users i.e. without the knowledge of the collection, and the retrieval environment, it is hard for the user to formulate queries that are aimed at well designed retrieval purposes. Thus the first run can be a trial to retrieve a few useful items from a given collection. These trial results can be examined for relevance and new formulations can be made to the queries which can result in retrieving improved additional items for the subsequent searches. This process can be manual or automatic. The manual/intellectual query reformulation where the task lies with the searcher it is possible for the system to take over this task entirely requiring only some yes-no answer from the user, this controlled automatic process which is convenient to use and effective is known as relevance feedback [5].

- Its aim is to improve the retrieved set by removing unwanted documents and adding more relevant documents without the user constructing new search strategies, and by using relevance or non-relevance information obtained from the user. The typical automatic relevance feedback operation involves

- An initial search with a user-supplied query and an initial retrieval of certain documents.
- Then, from a display (usually of titles or abstracts of the retrieved documents) the searcher identifies/chooses some relevant documents.
- These documents are used to modify the query by reweighting the existing query terms and/or by adding terms that appear useful and by deleting terms that do not.

This process creates a new query which resembles the relevant documents more than the original query does.

The main advantages of relevance feedback are

- Users do not need to know the details of the query formulation process i.e. knowledge of collection and search environment, but helps the user in constructing useful search statements
- The search operation is broken down into a sequence of smaller search steps which aim at approaching the desired area of subject without wandering
- Provides a controlled query alteration process designed to emphasize and deemphasize terms as and when required

Relevance feedback can be implemented in various ways depending on the retrieval technique used, e.g., vector space, probabilistic, etc., and also on the methods used to select terms for the feedback query. We can distinguish four term selection methods for query reformulation and query expansion

1. It relies entirely on the original query and uses only those terms in the new one
2. It uses terms from the original query and also adds terms from some other source, e.g., from all the adjacent terms in the maximum spanning tree (MST).
3. It is a mixed method because it uses combinations of the terms derived from the original query and from the documents retrieved and judged relevant as found
4. It abandons the terms from the original query and uses only terms found in the retrieved set of document

In all these cases, after the initial query formulation, the only form of feedback to the user is documents, and from the user are choices of documents.

The original process was designed to be used with vector queries of weighted search terms. A search expression is as given below

$$Q_0 = (q_1, q_2, \dots q_t)$$

Where q_i represents the weight of term i in query. The terms weights are either 0 or 1which represent an absent term and a fully weighted term respectively. The term can be chosen to be a concept, or a word/phrase, or a thesaurus entry. The relevance feedback generates a new vector

$$Q = (q_1, q_2, \dots, q_t)$$

Now the new terms are introduced by assigning a positive weight to terms with initial weight of 0, and old terms are deleted by reducing the previously positive weights to 0. Relevance feedback is easily implemented by graphically displaying the ranked lists of retrieved documents and screen pointers can be used to indicate relevant ones among them. These indications are then further used to construct modified queries.

3.1.1 Vector processing methods

In this feedback procedure both the documents and queries are represented as vectors of dimension t and in each of these, d_i and q_i represent the weight of term i in D and Q respectively. Thus the query-document similarity measure can be computed as the inner product of these vectors i.e.

$$\mathrm{sim}(D, Q) = \sum_{i=1}^{t} d_i \cdot q_i$$

Rocchio proposed an algorithm which gives us the best query leading to the retrieval of many relevant items from a collection of documents using

$$Q_{opt} = \frac{1}{n} \sum\nolimits_{relevant} \frac{D_i}{|D_i|} - \frac{1}{N-n} \sum\nolimits_{non-relevant} \frac{D_i}{|D_i|}$$

The D_i represents document vectors, and $|D_i|$ is the vector length. N is the collection size and n are the number of relevant documents. This initially cannot be used in query formulation because the set of n is unknown. After the initial relevance is made, the sums of relevant and non-relevant items are replaced by sums of known relevant and known non-relevant items and also the original query terms are preserved to be added. If n_1 and n_2 are relevant and non-relevant items, an effective feedback query would be formulated as given below

$$Q_1 = Q_0 + \frac{1}{n_1} \sum_{\text{known relevant}} \frac{D_i}{|D_i|} - \frac{1}{n_2} \sum_{\text{known non-relevant}} \frac{D_i}{|D_i|}$$

Here Q_0 and Q_1 are the initial and first iteration queries respectively.

For even more generality, we can modify the formulation using multipliers α, β, γ as

$$Q_{i+1} = \alpha Q_i + \beta \sum_{\text{rel}} \frac{D_i}{|D_i|} - \gamma \sum_{\text{non-rel}} \frac{D_i}{|D_i|}$$

We evaluate the importance of these multipliers later.

3.1.2 Probabilistic feedback methods

Another way of relevance feedback where the documents are ranked in decreasing order of rank as per the expression

$$\log \frac{\Pr(x|\text{rel})}{\Pr(x|\text{nonrel})}$$

Where Pr(x|rel) and Pr(x|nonrel) are the probabilities that a relevant or non relevant items have vector representation x.

Terms are assigned independently to relevant and non-relevant documents, weights restricted to 0 and 1 are assigned. A query similarity value can be calculated according to the equation

$$\text{Sim}(D, Q) = \sum_{i-1}^{t} d_i \log \frac{p_i(1-u_i)}{u_i(1-p_i)} + \text{constants}$$

Here again $p_i = \Pr(x_i = 1|\text{relevant})$ and $u_i = \Pr(x_i = 1|\text{nonrelevant})$. These values are needed for all documents and there are different methods to estimate these quantities. For the initial search when information regarding document relevance is not known, we assume a constant value for all terms say 0.5 for p_i and u_i can be the proportion of documents that contain the term i that is n_i/N

Hence initial $\mathrm{sim}(D, Q) = \sum_{i=1}^{t} d_i \log \frac{N-n_i}{n_i}$

For the feedback searches we assume that the term distribution in initial set is same as the distribution for the complete set of relevant items.

The following table shows the occurrences of term i in a collection of N documents.

	Relevant items	Non-relevant items	All items
D_i=1	r_i	$n_i - r_i$	n_i
D_i=0	$R - r_i$	$N - R - n_i + r_i$	$N - n_i$
All items	R	N-R	N

Table 1: Term occurrences.

If R represents the total number of relevant retrieved items and r_i is the number of relevant retrieved that include terms i and n_i is the number of retrieved items with term i then

$P_i = \frac{r_i}{R}$ and $u_i = \frac{n_i - r_i}{N-R}$ and we get the new feedback form as

$$\text{feedback } \mathrm{sim}(D, Q) = \sum_{i=1}^{t} d_i \log \left(\frac{r_i}{R - r_i} \div \frac{n_i - r_i}{N - R - n_i + r_i} \right)$$

For R=1, r_i=0 we get certain problem as the logarithmic expression is reduced to 0 so we add an adjustment factor of 0.5, so $P_i = \frac{r_i + 0.5}{R+1}$ and $u_i = \frac{n_i - r_i + 0.5}{N-R+1}$. An alternate adjustment factor is $n_i - r_i/N - R$ when the number of relevant documents not yet retrieved is small.

The advantage of this method over the vector method is that the feedback process is directly related to the derived weight for query terms because the similarity function increases by the weighting factor for each term that matches in a document. Also the set of relevant retrieved items are not used for query adjustment in this method.

3.2 Relevance feedback evaluation

Typically the positions of all retrieved relevant and non-relevant documents are taken into consideration when calculating effectiveness. However, when manual relevance feedback is used, where documents are confirmed as either relevant or not, and this has an influence on the next iteration of queries, then the resulting ranking of documents is affected by the user judgments. Depending on the effectiveness of the feedback mechanism, documents confirmed to be relevant are

Ranked before any other documents and documents that are confirmed to be non-relevant are either ranked very low, or not ranked at all, if not all documents are ranked. This effect artificially inflates evaluation measurements. It is desirable that only documents that are not assessed – the unseen documents – are used for evaluation of a feedback mechanism. Chang et al. (1971) offer options to control for this effect. The first is called modified freezing. It is a modification of the freezing method (full freezing), where the ranks of all documents assessed so far are frozen and only unseen documents are re-ranked. In modified freezing, only the ranks of documents up to the lowest ranked relevant document are frozen. A problem with this approach is that at later iterations, an increasingly large number of the ranking is frozen and that the effectiveness of the relevance feedback mechanism can seem worse than it actually is.

The evaluation should distinguish the true feedback effect from the artificial ranking effect as retrieved relevant documents will be used for feedback again with a much improved retrieval rank. Since an already seen item is of not much use to the user's satisfaction, the relevance feedback must be judged by its ability to retrieve new unseen relevant items.

A second option is residual ranking, where documents that are used for relevance feedback are removed from the collection before ranking with the reformulated query. A problem here is that eventually all relevant documents will be eliminated from the collection, which has an undesirable impact on evaluation measurements.
To evaluate the effectiveness of relevance feedback the two main measures used are recall and precision. Recall is defined as the proportion of relevant items that are retrieved from the collection, and precision is the proportion of retrieved items that are relevant.

There are twelve methods of relevance feedback for evaluation purposes which include six vector type modification methods and six runs of probabilistic feedback.

Given below are characteristics of six of these methods but first we describe each of them.

- Vector adjustment (Ide dec-hi) : add document term weights directly to query terms, use all relevant retrieved for feedback except the top most non-relevant items

$$Q_{new} = Q_{old} + \sum_{\text{all relevant}} D_i - \sum_{\text{one non-relevant}} D_i$$

- Vector adjustment (Ide regular) : add actual document term weights to query terms, use all the previously retrieved relevant and non-relevant items for feedback

  $Q_{new} = Q_{old} + \sum_{\text{all relevant}} D_i - \sum_{\text{all non-relevant}} D_i$

- Vector adjustment (standard rocchio) : add reduced term weights to query which follows division of term weights by number of documents used for retrieval, choose values of β, γ in range 0 to 1 such that β+γ=1

$$Q_{new} = Q_{old} + \beta \sum_{n_1 \text{reldocs}} \frac{D_i}{n_1} - \gamma \sum_{n_2 \text{non-reldocs}} \frac{D_i}{n_2}$$

- Probabilistic conventional :

$$Q_{new} = \log[p_i(1-u_i)/u_i(1-p_i)]$$

$$P_i = \frac{r_i+0.5}{R+1} \text{ and } u_i = \frac{n_i-r_i+0.5}{N-R+1}$$

- Probabilistic adjusted derivation :

$$Q_{new} = \log[p_i(1-u_i)/u_i(1-p_i)]$$

$$p_i = \frac{r_i + n_i/N}{R+1}$$

$$u_i = \frac{n_i - r_i + n_i/N}{N-R+1}$$

- Probabilistic adjusted derivation revised : same as above but for query terms use $\mathrm{r_i}$ and R instead of $\mathrm{r_i}$ and R where $\mathrm{r_i} = \mathrm{r_i} + 3$ and $\mathrm{R} = \mathrm{R} + 3$

In the first two methods of vector type as seen above, documents are added to original query vectors without normalization. In “dec-hi” all retrieved relevant and only one retrieved non-relevant is used and this single item notifies a point in the vector space from which it deviates. The “rocchio” uses reduced document weights to query modification. All the feedback methods produce weighted query terms. However feedback process does not specify the weights of the terms attached to the documents. A number of query expansion methods are applied in feedback process. First we contain the original query terms reweight them and use for feedback. In this type of expansion system, the terms with the highest frequency from previous retrievals are added to the original query and alternatively the highest weighted terms are used for query expansion.

We provide a table evaluating relevance feedback methods against five different collections (weighted documents, weighted queries), expanded by all terms

Relevance feedback method	Rank and avg precision	CACM(3204 docs 64 queries)	CISI(1460 docs 112 queries)	CRAN(1397 docs 225 queries)	INSPEC(12684 docs 84 queries)	MED (1033 docs 30 queries)
Initial run		.1459	.1184	.1156	.1368	.3346
Ide(dec hi)	Rank Precision Improvement	1 .2704 +86%	2 .1742 +47%	6 .3011 +60%	1 .2140 +56%	1 .6305 +88%
Ide(regular)	Rank Precision Improvement	7 .2241 +66%	18 .1550 +31%	15 .2508 +117%	4 .1936 +42%	2 .6228 +86%
Rocchio	Rank Precision improvement	2 .2552 +75%	39 .1404 +19%	8 .2955 +156%	14 .821 +33%	17 .5630 +68%
Probabilistic adjusted derivation	Rank Precision Improvement	11 .2289 +57%	36 .1436 +21%	3 .3108 +169%	32 .1621 +19%	5 .5972 +78%
Conventional probabilistic	Rank Precision Improvement	18 .2165 +48%	56 .1272 +7%	1 .3117 +170%	55 .1343 -2%	13 .5681 +70%

Table 2: Evaluating relevance feedback expanded by all terms.

Now we look at the table where in evaluation of feedback methods is done using expansion by most common words for the same five collections

Relevance feedback method	Rank and avg precision	CACM(3204 docs 64 queries)	CISI(1460 docs 112 queries)	CRAN(1397 docs 225 queries)	INSPEC(12684 docs 84 queries)	MED (1033 docs 30 queries)
Initial run		.1459	.1184	.1156	.1368	.3346
Ide(dec hi)	Rank Precision Improvement	4 .2479 +70%	1 .1924 +63%	13 .2498 +116%	2 .1976 +44%	3 .6218 +86%
Ide(regular)	Rank Precision Improvement	17 .2179 +49%	5 .1704 +44%	17 .2217 +92%	17 .1808 +32%	4 .5980 +79%
Rocchio	Rank Precision improvement	3 .2491 +71%	12 .1623 +37%	12 .2534 +119%	10 .1861 +36%	24 .5279 +55%
Probabilistic adjusted derivation	Rank Precision Improvement	14 .2224 +52%	10 .1634 +38%	18 .2120 +83%	9 .1876 +37%	14 .5643 +69%
Conventional probabilistic	Rank Precision Improvement	12 .2232 +53%	4 .1715 +45%	11 .2538 +120%	19 .1782 +30%	8 .5863 +75%

Table 3: evaluating relevance feedback expanded by common terms.

Weighted terms produce better results in a feedback environment. The comparison between above two tables' show that full query expansion is preferred over the expansion which is restricted (i.e. expansion by common terms) but the difference is reasonable that the expansion by common terms can be used when there are limitations in storage and processing times.

The vector processing model publishes ranked results in decreasing order of query-document similarity and thus it becomes easy to choose the first non relevant item from the list for the feedback purpose. This makes the "Ide dec hi" method as the best overall relevance feedback method as terms are added directly to query with only one non relevant item which makes it computationally efficient.

The probabilistic methods are not as effective as vector methods because of more computations and of the above methods, adjusted derivations was less effective.
The average length of original queries is important too because addition of terms affect the results as short queries gain more from the feedback process than collections with longer queries.

Thus we can conclude that relevance feedback proves to be an inexpensive method for reformulating queries based on already retrieved relevant and non relevant documents and this is generally incorporated in text retrieval systems.

3.3 Online relevance judgments

Once the initial query terms are selected, a search takes place and the results are displayed to the users for online relevance judgments to be obtained. Then there arises some questions; the question of which parts of the record relevance judgments should be based on, the online relevance assessment of the documents, and the size of the sample of relevant documents to be used for relevance feedback and query expansion.

In relevance feedback experiments the sample is defined at a cutoff level of the 10 or 20 top-ranked documents. These documents are then examined for relevance and those found relevant become the sample for the feedback iteration and the query expansion search. Relevance judgments are influenced by form, i.e. by the different document representations viewed, for example, title, citation, abstract and/or full text. The user judges it based on a binary value of relevance i.e., either “yes” or “no”.

Chapter 4: Query Expansion

The IR systems retrieval performance is improved by reformulating the initial queries by evaluating the user’s input and adding of search terms to expand the query for matching even more additional documents that are termed relevant. Expansion methods making use of local analysis have to go through the following steps: original query to rank an initial set of documents, all terms extracted from these are evaluated and ranked in order of their importance to query, top ranked terms are added to query, and with the reformulated query a final set of documents is ranked.

Query expansion can be of three types: manual, automatic and interactive. And each of these expansions can be based on either of the two types namely based on search results and based on knowledge structures.

The "curse of dimensionality" refers to the problem of selecting a set of search terms that can be effectively used to predict relevance. This problem arises because of the highly dimensional term space and to reduce the dimensionality we assume that the query terms act as good guides for predicting relevance. The association hypothesis states that if an index term is good at discriminating relevant from non relevant documents, then any closely associated index term is also likely to be good at this.

4.1 Manual query expansion

This is associated with Boolean online and CD-ROM searching. The most important and central aspect to online search is the search strategy development. Good search strategy development depends on the use of one's knowledge about online searching systems, indexing vocabularies and database construction, it also requires a good understanding of the mechanics of the matching paradigm of information retrieval and how this is implemented in the system searched. The result of such analysis will eventually determine the subsequent search formulation, which is the statement or set of statements which express the necessary query in a form understandable to the online system. It has to be decomposed correctly and the key concepts or facets have to be identified. Then the possible alternate actions depend on the choice of a particular strategy.

Some of the most commonly used strategies are the: building block, citation pearl growing, briefsearch, successive fractions, most specific facet first, or lowest postings facet first. In building blocks strategy all the terms belonging to the same concept are joined by Boolean OR operator and the results of each sub-search are joined by the Boolean AND operator. The Citation Pearl Growing strategy operates in a completely different manner. The searcher starts with a very direct search on the most specific term for each of the concept groups in the search request in order to find at least one citation. That is instead of OR-ing all the terms in each facet, as in the building block above, the searcher selects the most specific representative term of that facet. The single Boolean expression given below is known as the Briefsearch

$$(\text{term\_facet})_A \text{ AND } (\text{term\_facet})_B \text{ AND } (\text{term\_facet})_C \text{ AND } \ldots \text{AND } (\text{term\_facet})_N$$

Search strategy formulation is a highly unstructured problem and it requires a wide range of knowledge and moreover it cannot be automated. There is an array of tactics, heuristics and moves available to the searcher to choose from depending on the stage of the search. Tactics can be divided into groups according to the function they perform or according to the stages of the search at any one time. Moves deal with search situations where the retrieved set (a) is too large, (b) too small, or (c) off target. These are divided according to the searching style employed by the searchers. Heuristics are general rules of thought or action, mental operations, tactics, behaviors, or attitudes that tend to produce useful results in certain problem-solving situations.

4.2 Automatic query expansion (AQE)

This query expansion process is hidden in the overall retrieval process where in systems employ weighted or associative retrieval techniques. We have seen earlier that query expansion is based on searches and knowledge structures.

Based on search results

The query expansion is based on normalized vectors where both queries and documents are represented by weighted vectors. With a given cut-off point term vectors were added or subtracted depending on relevance feedback. Rocchio adjusted the method by allowing terms to be included in the expanded query if they were in the initial query or occurred in at least half the relevant documents. This adjustment provided positive results. SALTON & BUCKLEY evaluated twelve vector space and probabilistic feedback approaches across six test collections. The tests also involved two levels of query expansion.

Dillon and Desper have described an algorithm for automatically incorporating search terms into a query using a form of relevance weighting known as prevalence weighting. Positive and negative prevalence is computed based on the occurrence of terms in relevant and non-relevant documents which are retrieved from the initial search query. Depending on the prevalence weights, a number of threshold values for the prevalence weights exist and hence groups of terms are assigned to a particular category. A new Boolean query is constructed by OR-ing together groups of terms according to their position in the prevalence category. Terms in the highest prevalence category are added (OR-ed) as single terms. Terms from the second highest category are AND-ed as pairs of terms and so on. Finally bad (negative weight) terms are employed and these are NOT-ed. Any document containing one of these terms is not retrieved.

The OKAPI experimental online catalog uses a dictionary table of substitution terms. OKAPI expands a query by selecting the best terms from a list containing the original query terms together with terms extracted from all the records which the user has judged relevant. Terms are weighted using a scheme based on the F4 formula and which gives a higher weight to terms that occur in more of the relevant document and a lower weight to those that do not. The list of terms is then sorted by descending term weight.

Based on knowledge structures (collection dependent)

There are several areas on which work was done

- Term clustering - experiments explored a number of different clustering strategies and found that the effect of all these strategies are almost same. Retrieval effectiveness improved by term clustering.

- Term co-occurrence - since query terms tend to have high collection frequencies, their Nearest Neighbors, which are usually the terms added to the search by the expansion method, are also likely to have high collection frequencies
- Association thesaurus - is a matrix that consists of term-term similarities. It is based on how the terms in the collection are indexed i.e., for each term there is document vector space. The domain knowledge contained in a similarity thesaurus is then used to find additional search terms that are most similar to the entire query, rather than to select terms that are similar to a single term in the query.
- Conflation – based on stemming and string similarity measures

Based on knowledge structures (collection independent)

The automatic query expansion in the OKAPI online catalog in addition to using terms from the relevant retrieved records it also uses the classification codes that are assigned to the record.

4.3 Interactive query expansion (IQE)

In the reformulation process, the users are presented with the search terms. In interactive query expansion as opposed to automatic query expansion there are two things responsible for determining and selecting terms for expansion. One is the retrieval system itself which, like the automatic query expansion, is designed to select terms and then weigh and rank them accordingly. The other is the user, who is presented with the ranked list of terms and has to decide which terms to be added to the search. As it is based entirely on the user's preferences, it is the user's responsibility to determine the final terms and it becomes increasingly difficult to point out the reasons for success or failure because of the uncontrollable variables.

Based on search results

The system presents to the user a list of terms based on their occurrences in an identified set of documents. Then the user feeds back the choice of terms. The document set on which this analysis is based may either be simply a set retrieved in the usual way, or it may consist of documents individually selected as relevant by the user.

The ZOOM feature on ESA/IRS is a tool for online searching along these lines. It performs term frequency analysis on a number of records from the retrieved set(s). The user is then presented with screen-displays which contain terms in a frequency ranked order. The searcher selects terms which then can use to expand the query. ESA/IRS also offers QUESTQUORUM as a simple interface which can do a semi-automatic query expansion based on terms selected by the user from a ZOOM-like display.

There are some other systems which have used the term frequency analysis function in some form or the other. CITE, USERLINK, IT, and OASIS uses the user feedback also in a similar manner. It automatically performs term frequency analysis on the records marked as relevant, and then it presents the terms in ranked order to the user for selection. The EXPLORE command is used to achieve the query expansion. There are several other commands to view and edit the results/lists.

Interactive query expansion was investigated from two different perspectives, (a) studying end users during the process of query expansion, especially the term selection process, and (b) studying the behavior of ranking algorithms for query expansion. The overall search results provided some evidence for the effectiveness of interactive query expansion. The user-based aspect of the research investigated the processes of interactive query expansion, term selection for query expansion by users, and the user perception, understanding, and assignment of term relationships from a knowledge structure.

Based on knowledge structures (collection dependent)

The Examples of interactive query expansion based on the collection are the EXPAND or ROOT commands available in online vendors. These provide a form of feedback from a knowledge structure of the database which is the dictionary file. Users are given an alphabetical listing of descriptors and free-text terms to choose from and expand or modify their query.

EUREKA is an experimental full text retrieval system which uses a user specific thesaurus. Each user can create and maintain a personal thesaurus which is used by EUREKA at search time to find synonyms for the query terms. As additional user aids EUREKA can present on demand either a histogram of term frequencies based on the retrieved documents, or word-lists of terms that are used in many documents or have high average frequencies. From these lists the user selects terms to refine the retrieved set.

Based on knowledge structures (collection independent)

The query expansion examples are found in AID, CITE, and some expert systems which we are going to mention.

The Associative Interactive Dictionary (AID) is a prototype system developed for the Medline and Toxline databases at the National Library of Medicine. It automatically generates and displays related terms, synonyms, broader and narrower terms and other semantic associations for a given search term. The terms are derived from titles, abstracts and/or controlled indexing fields from retrieved documents. These terms are displayed in ranked order according to a `relatedness' value (R) which is calculated using a modified chi-square value. The retrieved set is defined as the set of documents retrieved by a given search term or Boolean query. AID operates by storing a subset of the inverted files for the two databases in its in-core hash table. The hash table terms represent all the inverted files' index terms with a frequency of four or more postings. Searches are carried out in the usual Boolean fashion and AID can be implemented at any time through the EXECUTE command.

CITE is used by the searcher who enters an enquiry statement in natural language. It parses the input, identifies spelling mistakes, requests their clarification and then suggests to the searcher a set of potentially applicable single words which are ranked by some weighting formula.

All expert systems which have pre-search aid modules, such as CANSEARCH, CONIT, TOMESEARCHER, etc., use a knowledge structure independent of the collection and help in the suggestion of terms.

4.4 Ranking algorithms and term selections for query expansion

In both automatic and interactive approaches the ranked order of terms is of primary importance. The order should preferably be one in which the terms that are most likely to be useful are close to the top of the list. We know from IR research with respect to the relationship that holds between term frequency and term value and the effect on retrieval is that very frequent terms are not very useful; middle frequency terms are quite useful; infrequent terms are likely to be useful but not as much as the middle frequency terms; very infrequent terms are useful terms in the sense that when present they are good indicators of relevance. From this knowledge it can, therefore, be hypothesized that a good term ranking algorithm [4] would bring the middle frequency terms near the top of the list. Some of the ranking algorithms are

The F4 algorithm

The theory of relevance weights uses the basis of relevance information for weighting of query terms. Term independence and document ordering assumptions are made and the basic formula is

$$w_t = \log \frac{p_t(1-q_t)}{q_t(1-p_t)}$$

Where p_t is the probability of term t occurring in a relevant document, and q_t is the probability of term occurring in a non relevant document. We know the estimates of these probabilities as $p_t = r/R$ and $q_t = (n-r)/(N-R)$ where N is the total number of documents in the collection, R is the sample of relevant documents, n is the number of documents indexed by term t and r is the number of relevant documents assigned to term t.

$$w_t = \log \frac{\frac{r}{R-r}}{\frac{n-r}{N-n-R+r}} = \log \frac{r(N-n-R+r)}{(n-r)(R-r)}$$

If any of the four braces in the equation is zero then it gives us infinite weights, to overcome this we modify the formula by adding 0.5 to each of the quantities and the result is known as point-5 formula

$$= \log \frac{(r+0.5)(N-n-R+r+0.5)}{(n-r+0.5)(R-r+0.5)}$$

The F4 modified algorithm

Robertson modified by adding new terms to the query.

$$w_t = \log \frac{(r+c)(N-n-R+r+1-c)}{(n-r+c)(R-r+1-c)}$$

Where c = n/N

In automatic query expansion every term from the relevant document would be weighted using above formula and added to the search. In interactive expansion the term weighting would be in same fashion by the user selection.

Porter's algorithm

Porter used the following rank formula

$$\text{porter} = \frac{r}{R} - \frac{n}{N}$$

It can be noticed that the weight is influenced by the term occurrence in the relevant document set as well as term frequency in the collection. The r/R portion never becomes zero there should be at least one document containing the term termed relevant and it can have a maximum value of 1 when r=R.

The EMIM algorithm

The expected mutual information measure uses relevance information in such a way that an assumption is made where the index terms may not be distributed independently of each other.

$$\mathrm{EMIM} = \mathrm{E_{iq}} = \sum_{\mathrm{t_i},\mathrm{w_q}} \Delta_{\mathrm{iq}}\, \mathrm{P}(\mathrm{t_i},\mathrm{w_q}) \log \frac{\mathrm{P}(\mathrm{t_i},\mathrm{w_q})}{\mathrm{P}(\mathrm{t_i})\mathrm{P}(\mathrm{w_q})}$$

Where t_i indicates the presence (1) or absence (0) of a term; w_q indicates that a document is relevant (1) or non-relevant (0); Δ_{iq} indicates the value of a term as a relevance discriminator and it is 1 if $t_i = w_q$ or -1 if unequal. The second term in the formula can be represented as D_{iq} degree of involvement and the last term is the probabilistic contribution.

The WPQ algorithm

In the query expansion stage of search an assumption is made where in we consider the statistical independence between query expansion terms and the terms in the previous search formulations. The presence or absence of query expansion terms doesn't affect the initial distribution. The inclusion of a term t in the search formulation will increase the retrieval effectiveness by

$$a_t = w_t(p_t - q_t)$$

Where w_t is the weighting function as given by F4 point-5. P is the probability of term occurring in a relevant document and Q is the probability of a term occurring in a non relevant document. This means that inclusion of a query expansion term should be based on the ranking of a_t.

$$a_t = \log\frac{(r+0.5)(N-n-R+r+0.5)}{(n-r+0.5)(R-r+0.5)} \cdot (\frac{r}{R} - \frac{n-r}{N-R})$$

The p_t-q_t component like the porter formula is influenced by the frequency of occurrence of a term in the relevant document set as well as term frequency in the collection.

The ZOOM term frequency ranking

ZOOM is a frequency analysis tool available in the ESA/IRS online vendor. It provides the automatic frequency analysis of phrases, single words, codes, or a combination of these contained in a selected set of references. Once a set of records is generated in a file the searcher may ZOOM the set. The ZOOM command can analyze up to 20,000 records. ZOOM processes the records in the set and the phrases and/or single words of the analysis are displayed in columns. All terms are ranked in descending order of their frequency of occurrence in the sample set. Within ties, i.e. whenever there is more than one term with the same frequency of occurrence, terms are ranked in alphabetical order.

4.4.1 Evaluation of the ranking algorithms

The effectiveness is measured through precision and recall. The methodology followed for the ranking of the terms of each search is: extract terms presented to users for each search, calculate weights for the terms with each of the above mentioned algorithms, divide each of the resulting ranked lists into parts, match the users choices of terms to each ranked list, for each list tally the distribution of all terms over each part.

We then study the top 5 ranked terms of each list. The difference between the 5 top ranked terms and the user termed 5 best terms is noted, these latter terms are used for query expansion. For further qualitative measures, we follow the given methodology: assigning ranks to terms in the ranked lists, determining the position of each of the 5 best terms, adding the rank positions for the 5 terms of each list, using the wilcoxon test to find the statistical significance, and calculating the Pearson co-efficient for pairs of algorithms.

There appears to be less difference between WPQ and EMIM and between F4 and F4 modified but overall there are significant differences in order. Porter algorithm has similar performance to WPQ and EMIM. These both algorithms have better ranking of user preferred terms for query expansion. By inspection of all the algorithms there can be a new algorithm that:

- Ranks terms according to the frequency of occurrence in the relevant document set
- Resolves ties according to the term frequency from low to high

4.5 Robustness of query expansion

The robustness [6] of an IR system has to be improved for handling the queries in an effective way. A system is said to be robust when it has achieved both a high Mean Average Precision (MAP) for the whole set of topics and a significant MAP value over some worst X topics (MAP(X)). As query expansion weakens the performance on worst topics, a selective application of QE is needed for a robust retrieval system. Global performance gives us the average behavior of the system. There are two evaluation measures for robustness defined in TREC, the number of topics with no relevant documents in the top retrieved 10 (denoted as NrTopicsWithNoRel) and MAP(X) which is used to measure the area under the average precision over the worst X topics. The problem of finding out poor performing query is known as query-difficulty or query specificity.

Methodologies have been developed to improve the performance on worst topics for robust QE activation. The framework is based on term weighting models. The DFR (divergence from randomness) within-document term weighting models are:

I(n)OL2, I(n_e)OL2, I(n)B2, I(n_e)B2, I(n_e)OB2. These are obtained from the following formula

$$\mathrm{Info}_{\mathrm{DFR}} = -\log_2 \mathrm{Prob}(\mathrm{term\_freq} | \mathrm{doc}_{\mathrm{freq}}, \mathrm{Freq}(\mathrm{term}_{\mathrm{collection}}))$$

Where Prob is the probability of finding a within-document term-frequency. This formula is normalized by finding the probability only in the set of documents containing the term. So the final weighting formulas are given below:

I(n)OL2: $\frac{\mathrm{tfn}}{\mathrm{tfn}+1}\log_2(\frac{\mathrm{N}-\mathrm{doc}_{\mathrm{freq}}+1}{\mathrm{doc}_{\mathrm{freq}}+0.5})$

I(n_e)OL2: $\frac{\mathrm{tfn}}{\mathrm{tfn}+1}\log_2(\frac{\mathrm{N}-\mathrm{n}_\mathrm{e}+1}{\mathrm{n}_\mathrm{e}+0.5})$

I(n)B2: $\frac{\mathrm{Freq(term|collection)}+1}{\mathrm{doc}_{\mathrm{freq}}\cdot(\mathrm{tfn}+1)}(\mathrm{tfn}\cdot\log_2\left(\frac{\mathrm{N}+1}{\mathrm{doc}_{\mathrm{freq}}+0.5}\right))$

I(n_e)B2: $\frac{\mathrm{Freq(term|collection)}+1}{\mathrm{doc}_{\mathrm{freq}}\cdot(\mathrm{tfn}+1)}(\mathrm{tfn}\cdot\log_2\left(\frac{\mathrm{N}+1}{\mathrm{n}_\mathrm{e}+0.5}\right))$

I(n_e)OB2: $\frac{\mathrm{Freq(term|collection)}+1}{\mathrm{doc}_{\mathrm{freq}}\cdot(\mathrm{tfn}+1)}(\mathrm{tfn}\cdot\log_2\left(\frac{\mathrm{N}-\mathrm{n}_\mathrm{e}+1}{\mathrm{n}_\mathrm{e}+0.5}\right))$

Where again

$$\mathrm{tfn} = \mathrm{term}_{\mathrm{freq}}\cdot\log_2(1+\mathrm{c}\cdot\frac{\mathrm{average}_{\mathrm{document}_{\mathrm{length}}}}{\mathrm{document}_{\mathrm{length}}})$$

N is size of collection

$$\mathrm{n}_\mathrm{e} = \mathrm{N}\cdot(1-(\frac{1}{\mathrm{N}})^{\mathrm{Freq(term|collection)}}$$

Freq(term|collection) is the within-collection term-frequency, term_freq is the within-document term-frequency, doc_freq is the document-frequency of term, and c is set to 3.

The weight of the expanded query term q^* is given as:

$$\text{weight}(\text{term} \in q^*) = \text{tfq}_n + \beta \cdot \frac{\text{info}_{\text{DFR}}}{\text{MaxInfo}}$$

Where tfq_n is the normalized term-frequency within the original query, $\text{MaxInfo} = \arg_{t \in q^*} \max \text{info}_{\text{DFR}}$, where info_{DFR} is a term frequency in the expanded query given by formula

$$\text{Info}_{\text{DFR}} = -\log_2 \text{Prob}(\text{Freq}(\text{term}|\text{TopDocuments})|\text{Freq}(\text{term}|\text{collection}))$$

The term weighting models compute the probability of obtaining a within-document term-frequency whereas the within-query term-frequency computes the probability of obtaining a given term-frequency within the top most retrieved documents.

Parameters with C=3	I(n)B2	I(n_e)B2	I(n)OL2	I(n_e)OL2	I(n_e)OB2
P@10	0.4180	0.4070	0.4130	0.398	0.3940
MAP	0.2434	0.2503	0.2519	0.2479	0.2329
Top10withNoRel	18	18	17	20	11
MAP(X)	0.0084	0.0065	0.0077	0.0058	0.0096

Table 4: robustness evaluation.

The table compares a baseline run with the full QE runs. The I(n_e)OB2 is the baseline as it performs better on most difficult topics, this unexpanded run achieves the best MAP(X) and the lowest NrTopicsWithNoRel.

The QE effectiveness is related to the number of documents which are relevant for a query in the set of top most ranked documents. If the precision of first-pass is high then there are good chances of extracting additional terms.

4.6 Techniques for efficient query expansion

Query expansion is used to significantly improve the retrieval effectiveness and for the effective method of expanding queries local analysis is useful. These methods determine the top ranked documents from which additional query terms are extracted and the drawback for such is increasing costs during query evaluation. Surrogates built from past queries require large query logs so an effective way is to use brief summaries, a pool of most important terms for each document.

Terms which are weighed obtained from judged documents are added to original query which are then reissued to rank the remaining relevant documents. IQE increases effectiveness although AQE is likely to give a better performance on an average. In AQE, query terms are added from highly ranked documents and an alternative is to construct similarity thesauri ahead of time which can be accessed at query time. In general the use of thesauri has not been of much success but the combined approach can be successful at times.

The Okapi BM25 measure is an effective method for query expansion.

$$\mathrm{bm25}(q,d) = \sum_{t \epsilon q} \log \frac{N - f_t + 0.5}{f_t + 0.5} \times \frac{(k_1 + 1) f_{d,t}}{k + f_{d,t}}$$

Where terms t appear in query q; the collection N contains d documents; f_t documents contain a particular term and a particular document contains a particular term $f_{d,t}$ times; k is $k_1((1 - b) + b \times L_d/AL$; constants k and b are set to 1.2 and 0.75 respectively; L_d and AL are document length and average document length respectively.

The expansion method proposed by Robertson and walker where E terms with the lowest term selection value (tsv) are chosen from the top R ranked documents

$$\mathrm{TSV_t} = \left(\frac{\mathrm{f_t}}{\mathrm{N}}\right)^{\mathrm{r_t}} \frac{\mathrm{R}}{\mathrm{r_t}}$$

Where a term t is contained in $\mathrm{r_t}$ of the top R ranked documents. The expansion terms are added to the original query but instead of using their Okapi value, the weights are calculated by the formula:

$$\mathrm{w} = \frac{1}{3} \times \log\left(\frac{(\mathrm{r_t} + 0.5)/(\mathrm{R} - \mathrm{r_t} + 0.5)}{(\mathrm{f_t} - \mathrm{r_t} + 0.5)/(\mathrm{N} - \mathrm{f_t} - \mathrm{R} + \mathrm{r_t} + 0.5)}\right)$$

The standard values of E and R are chosen to be 25 and 10 respectively.

We have seen the five stages of expansion methods using local analysis, now we look at the scope of gaining efficiency for each stage [7].

During the initial ranking stage where for each query term an inverted list which is retrieved has to be accessed and the costs are directly proportional to the size. Way of cutting the cost would be to store the documents by the order of impact the terms have rather than storing all the documents during indexing. Surrogates can be used for documents but still the full index is needed for the final ranking.

During the fetching of documents from highly ranked ones, surrogates which are a fraction of the size of documents can be retrieved that provide a pool of expansion terms. The reduced sizes improve efficiency by reduced cache misses and smaller seek times.

During the extraction of candidate terms, instead of parsing the full text documents the in-memory surrogates are used which are pre-parsed and pre-stopped. These are pointers which reference terms and identify inverted lists and thus are smaller than terms.

During the selection of expansion terms the information of TSV is cached in the memory and can thus provide faster and fewer terms for selection.

During the final ranking, we approach it in similar way as the first phase using impact-ordering. All these methods reduce the costs and we consider some more methods of improving efficiency for QE as shown below [8]

- Reducing collection size for sourcing expansion terms: In large collections there are multiple documents on the same topic as of the query and it would be wise to access documents sampled at random but still representing the overall collection. Use of centroid clusters and documents stored in pre-parsed format are tested.
- In-memory document summaries: a small auxiliary database can be cached which stores the summaries of documents. They are the terms with the highest tf·idf values. Then summaries can be built in two ways; one is to have a fixed number of highly-ranked terms per document and the other way is to choose a threshold value and all the terms having tf·idf greater than that are in the summary. During querying surrogate terms ranked against original query are used for selection.

- During the ranking process, as original query terms get processed twice it is better to process the expansion terms without clearing the accumulator table which was used for initial ranking. As most expansion terms have high idf it is important to process them before the original query terms which have lower values.
- Query associations: In this method [3] we select expansion terms from past user queries that are associated with the collection. As query logs are available associations become effective but a minor disadvantage would be that an extra index needs to be referenced while evaluation. On the flip side the advantages of association are that they are pre-stemmed stored in a parsed from and hence easier to retrieve. The query associations provide a good summary of the document giving a matching description of the content. Past queries are used to form affinity pools from which expansion terms can be selected. This pools can be formed by the process as described; for the queries that are to be expanded, up to three past queries that are similar to present are identified and the top 100 documents that are returned for each past query are merged to form a pool from which candidate terms are selected after running the original query against it and terms are selected using TF-IDF scores which will be described later in term-weighting approaches. This technique improves average precision by around 15% according to Fitzpatrick and Dent.

Query association (Scholer & Williams) is a process where in user queries become associated with a document if they have high statistical similarity with it. Once a query is submitted to a system, similarity score using Okapi is calculated and this query becomes associated with the top N documents that are returned. If an upper bound, say M is imposed on the number of queries that are associated with a single document, we can get efficiency by replacing the least similar document with a new one. The document summaries aid the users in judging the relevancy.

Apart from past queries there are other ways of forming document surrogates and one such way is use of anchor text (Craswell et al.). In this, the text content of hyperlink texts or anchor tags link that are linked to a document are extracted to form surrogates for finding entry pages to a website. These are significantly effective than full text retrieval but not useful in topic-finding tasks.

If ranking and term-selection are considered to be the two steps in a generalized expansion we get four schemas from the framework

- FULL-FULL : if we use a single collection for all steps which are based on the full text of documents in the collection
- FULL-ASSOC : original query run on full text collection after which the top expansion terms are selected from the set of queries that have been previously associated with the top documents
- ASSOC-FULL : initially rank directly on the surrogates built from associations and then choosing terms from the original documents

- ASSOC-ASSOC : rank the document surrogates built from associations and then choose the terms from the top ranked surrogates itself.

Alternate way to find terms from past user queries is to treat them as documents. We then source expansion terms by ranking the individual queries and selecting terms from the top past queries returned. These have no direct relation with any particular full text documents in the collection and this schema is called as query-query.

Another source of terms for query expansion is anchor texts which have a direct link with the documents in the collection. We select the terms from the top anchor text surrogates and then search the surrogates again using the expanded query. This schema is called as LINK-LINK.

Performance of expansion techniques of TREC-10 queries on TREC-10 collection

Type	Avg P	P@10	P@20	P@30
Base	0.1487	0.2714	0.2235	0.2000
Assoc-assoc	0.1893	0.3249	0.2888	0.2204
Assoc-full	0.1820	0.3184	0.2796	0.2222
Full-full	0.1584	0.2796	0.2571	0.2333
Query-query	0.1567	0.2755	0.2357	0.2116
Full-assoc	0.1549	0.2571	0.2276	0.2068
Link-link	0.1454	0.2653	0.2153	0.1905

Table 5: query associations schemas.

In this table we have compared baseline full text retrieval with the different schemas based on precision metrics. These are ordered by decreasing average precision. From this we can conclude that query association is an effective tool in initial querying stage prior to expansion.

4.7 Term weighting formulae in text retrieval

In automatic text retrieval, words extracted from texts of documents and queries are used for content matching. As we know the documents and queries are represented by term vectors, a typical query can be of the form given below

$Q= (q_a \text{ and } q_b) \text{ or } (q_c \text{ and } q_d) \text{ or } ....$

The term vectors for document is as given below

$$D = (t_0, w_{d_0}; t_1, w_{d_1}; \dots; t_t, w_{d_t})$$

Weight w_{dk} is equal to 0 when term k is not assigned to document or else equals 1 and since these weights are restricted the vector product to calculate similarity measures the terms that are jointly assigned to query and document. In some cases normalized weight assignments are used where the individual term weights depend on weights of other terms in the same vector.

The term weight using vector length normalization factor is $\frac{w_{dk}}{\sqrt{\sum_{\text{vector}} w_{di}{}^2}}$

A vector matching system provides ranked retrieval output in decreasing order of similarities. Over the years it has been observed that single terms for document content identification is not enough and this has led to the generation of sets of terms like related terms, term phrases, thesauri and knowledge bases. The assumption is that words that co-occur with some general frequencies are related to each other in a collection. Most automatically derived terms dependencies are valid only in the local documents from which the original term groups were extracted. If single terms are used for content identification then there must be some kind of descriptors which distinguish between individual terms and this has led to the concept of use of term weights.

The main function of this term-weighting system [1] is to improve the retrieval effectiveness based on precision and recall. Systems are preferred that have high recall by retrieving more relevant things and high precision by rejecting the items that are irrelevant.

There are three main term weighting factors that enhance recall and precision

- Term frequency (tf) – terms that are frequently mentioned in documents appear to enhance recall measuring the frequency of occurrence of terms.
- Inverse document frequency (idf) – since frequency alone cannot ensure retrieval effectiveness where the high frequency words are scattered throughout the collection then there are chances of retrieving all documents that decrease the precision. Thus this factor varies inversely with the number of documents n to which a term is assigned in a collection of N, computed as $\log N/n$.

- Normalization factor - it is useful in systems with varying vector lengths. Since large documents have large vectors the possibility of matching increases which may again reduce the effectiveness of precision hence normalization is required to equalize the lengths.

A measure of term importance may be obtained by using the product of tf and idf since best terms are those which have high frequencies but low overall collection frequencies.

In probabilistic models term relevance weight is defined as the proportion of relevant documents in which a term occurs divided by the proportion of non relevant items in which the term occurs.

A number of experiments have been described with the combination of these components. In these experiments, each term-weight combination is expressed using two triplets of the above 3 components for both the document (first triplet) and the query (second triplet).

Term weighting components:

Term frequency component		
b	1.0	binary weight equal to 1 for terms present in a vector
t	tf	raw term frequency
n	$0.5 + 0.5\ \frac{\mathrm{tf}}{\max \mathrm{tf}}$	lies between 0.5 and 1
collection frequency component		
x	1.0	no change in weight; use original b, t, or n
f	$\log\frac{N}{n}$	multiply tf factor by inverse collection frequency factor
p	$\log\frac{N-n}{n}$	multiply tf by probabilistic inverse frequency factor
normalization component		
x	1.0	no change
c	$1/\sqrt{\sum_{\mathrm{vector}} w_i^2}$	use cosine normalization where each term weight is

divided by factor representing vector length

Table 6: term weighting components.

Term weighting formulas:

Weighting system	Document term weight	Query term weight
Best fully weighted system(tfc.nfx)	$\frac{tf.\log\frac{N}{n}}{\sqrt{\sum_{vector}\left(tf.\log\frac{N}{n_i}\right)^2}}$	$\left(0.5+\frac{0.5tf}{\max tf}\right)\cdot\log\frac{N}{n}$

Best weighted probabilistic (nxx.bpx)	$0.5 + \frac{0.5tf}{\max tf}$	$\log \frac{N-n}{n}$
Classical idf weight (bfx.bfx)	$\log \frac{N}{n}$	$\log \frac{N}{n}$
Binary term independence (bxx.bpx)	1	$\log \frac{N-n}{n}$
Standard tf weight(txc.txx)	$\frac{tf}{\sqrt{\sum_{vector}(tf_i)^2}}$	Tf
Coordination level(bxx.bxx)	1	1

Table 7: term weighting formulas.

Performance results for the above methods over 5 collections

Term weighting methods	Rank and avg precision	CACM	CISI	CRAN	INSPEC	MED	Avg of these
Best-fully weighted(tfc.nfx)	Rank P	1 0.3630	14 0.2189	19 0.3841	3 0.2626	19 0.5628	11.2
Weighted-with inverse freq unused(txc.nfx)	Rank P	25 0.3252	14 0.2189	7 0.3950	4 0.2626	32 0.5542	16.4
Classical tf.idf(tfx.tfx)	Rank P	29 0.3248	22 0.2166	219 0.2991	45 0.2365	132 0.5177	84.4
Best-weighted probabilistic(nxx.bpx)	Rank P	55 0.3090	208 0.1441	11 0.3899	97 0.2093	60 0.5449	86.2
Classical idf (bfx.bfx)	Rank P	143 0.2535	247 0.1410	183 0.3184	160 0.1781	178 0.5062	182
Binary independence probabilistic(bxx.bpx)	Rank P	166 0.2376	262 0.1233	154 0.3266	195 0.1563	147 0.5116	159

Standard weights cosine normalization(txc.txx)	Rank P	178 0.2102	173 0.1539	137 0.3408	187 0.1620	246 0.4641	184
Coordination level binary vectors(bxx.bxx)	Rank P	196 0.1848	284 0.1033	280 0.2414	258 0.0944	281 0.4132	260

Table 8: performance measures for term weighting.

Conclusions:

1) Methods 1 and 2 produce comparable performance for all collections and are recommended for natural language texts and abstracts
2) Method 3 is poor for collections CRAN and MED where very short queries are used with little deviation in the query length
3) Method 4 is the best of the probabilistic weighting systems and less effective than the enhanced weighting methods of 1 and 2
4) Methods 5 to 7 are not that effective for all the collections
5) The coordination level matching of binary vectors is perhaps one of the worst possible retrieval strategies.

Other conclusions are:

- Long query vectors require a greater discrimination among query terms based on term occurrence frequencies
- Factor f is similar to factor p
- Query normalization doesn't affect query document ranking or the performance

Therefore,

Best document weighting tfc, nfc (or tpc, npc)

Best query weighting nfx, tfx, bfx (or npx, tpx, bpx)

Chapter 5: Different kinds of QE

5.1 Query expansion with auxiliary data structures

The standard ranking techniques return documents that contain same term as the query while identification of some relevant documents require finding of alternate query terms. Global analysis depends on term co-occurrence and is not necessarily query dependent. A new method [2] was proposed that draws candidate terms from brief document summaries that are held in memory for each document. While approximately maintaining the effectiveness of the conventional approach, this method significantly reduces the time required for query expansion by a factor of 5-10.

The graph below shows a drop in average precision if summaries consist only of terms with extremely low tf.idf values, of below 0.25. This would suggest that average precision could be optimized by using only terms that have a tf.idf value which falls into the band between 0.25 and 1.25 for TREC 8.

They considered two options expansion via reduced size collection and via document surrogates. The tf.idf summaries were successful because they are smaller than the original collection.

Varying average precision and associated memory cost with the number, cutoff value of summary terms and percentage of document used for summaries, respectively. Use of the TREC 8 collection and queries.

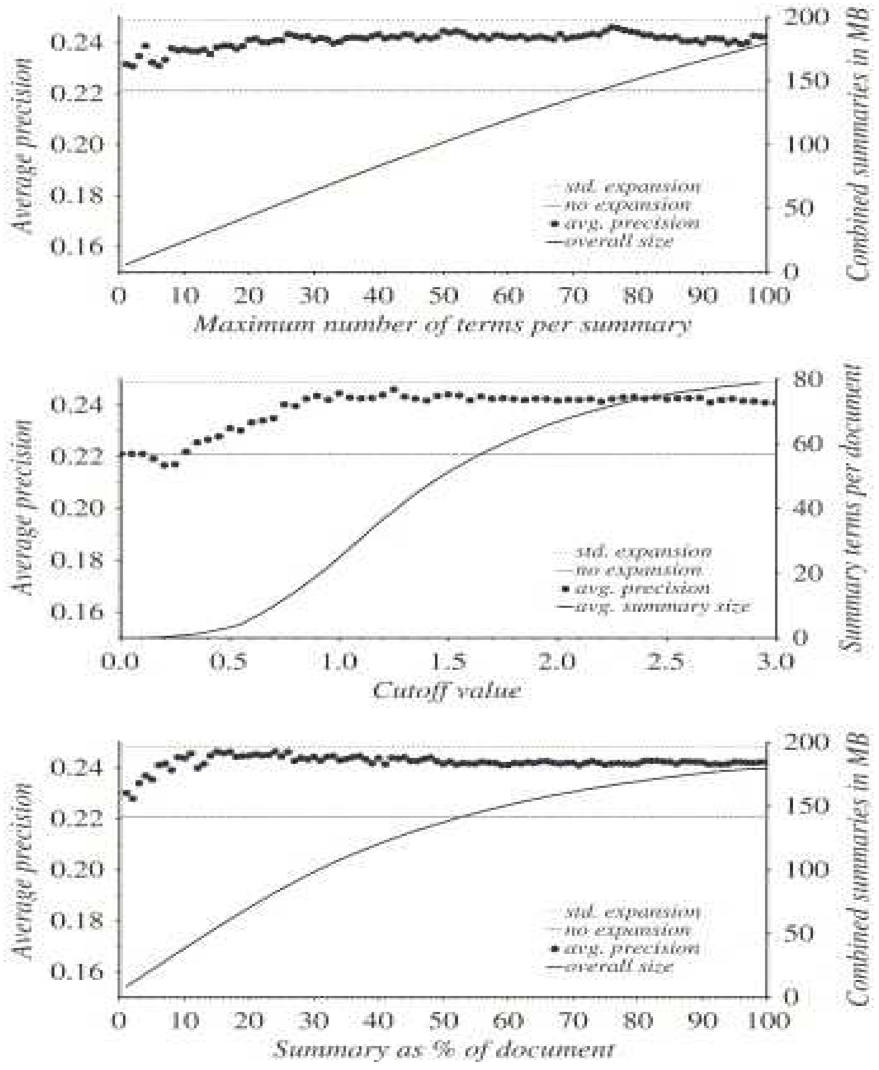

Figure 1: TREC 8 collection performance graph.

5. 2 Query expansion using topic and location

The approach [10] combines exploring both the location and topic information. Users at different locations may have different vocabularies for the same specific topic and hence they may use different query terms. This is known as query location sensitivity. Huang et al have proposed a hierarchical model to classify query terms at two different levels (location sensitive versus location non-sensitive, then same location sensitive versus different location sensitive). During the experimentation IP addresses were used to locate users. Experiments show that: the location based query expansion improves the search results significantly for the location sensitive queries; the precision of the query classification model is more than 80%; and location and topic based approach is significantly better than other query expansion approaches, especially on general webpage search. Earlier algorithms with conventional probabilistic retrieval approach are document based (Arasu et al. 2001).With this approach, an initial query is executed and a set of documents are returned. Then a set of terms are obtained from the top relevant documents, which are combined with the initial query to generate and return a more relevant set of documents. Cai et al propose a method based on the divergence of the query, which calculates the relevance of queries according to their distribution in documents (Cai, van Rijsbergen, & Jose 2001). Also probabilistic models, such as Markov Chains, are applied to improve the performance by combining different methods at successive stages (Collins-Thompson & Callan 2005).

By clustering the documents to different topics, they scaled down the document relevance to the topic relevance, and used the topic relevance to identify the similarity between queries. In addition they made use of the location information to determine whether the query is location sensitive and which type of query expansion should be applied. By comparing the improvements on Citeseer data and Excite data, they observed that the query location sensitivity is much more obvious in general webpages than in academic documents.

5.3 Query expansion using lexical-semantic analysis

The small collections with single-domain thesauri can reduce the mismatching problem of vocabularies while expansion. Concepts are represented by WordNet synonym sets (synsets). Source terms derived from lexical aids have improved performance but expansion by broad terms from hierarchical thesauri was found to be inconsistent. In this study [11], queries were examined using the relations encoded in WordNet, a large lexical system built at Princeton University. Concepts are the listed words that pertain to the topic which are marked as relevant. The expansion process is parameterized by setting the length of synsets to a particular length for each run. Stems added through different lexical relations are kept using extended vector space model.

Algorithm to automatically select sysnets for expansion

```
for (each query word w)
{ if (w not already expanded and document frequency of w < N )
  {
      expand all synsets containing w producing kin list of w
  }
 }
for (each relative in the set of kin lists)
{
if (relative occurs in more than 1 list)
add relative to query vector
}
```

5.4 Query expansion using random walk models

Term relations are an important aspect of information retrieval. They have described a Markov chain framework that combines multiple sources of knowledge on term associations [12]. The stationary distribution of the model is used to obtain probability estimates that a potential expansion term reflects aspects of the original query i.e. A query is modeled as a combination of aspects, and expansion terms are favored that are not only more rare relative to the collection, but also semantically close to multiple query aspects.

5.5 Probabilistic query expansion using query logs

There is a large amount of information recorded in query logs during web interaction and this is the idea implemented to find probabilistic co-relations between query terms and document terms [13]. Each session consists of a query and a set of documents that the user has clicked which makes it more reliable than pseudo relevance feedback. The query expansion reflects the user preferences at that specific time. For every query term, all co-related document terms are selected based on conditional probability. By combining probabilities of all queries, we calculate cohesion weight of document term for new query. Thus for every query, there is a list of weighted candidate expansion terms. The top ranked terms can be selected.

5.6 Query expansion using Apriori Algorithm

The proposal of using association rule discovery to find the candidate terms and enhance the queries is the basic idea for this type of expansion. Apriori algorithm is one such association rule discovery used in data mining to extract useful data from a large database. To apply association rule mining, each document can be viewed as a transaction with each word representing an item. They have achieved an improvement of 19% without the help of thesauri or any user intervention [14].

Chapter 6: conclusion

This report has discussed various types of query expansion which has been studied along with the various ways of retrieving terms and marking them as relevant. Each and every experiment conducted had its usefulness and limitations, and there is still a lot of scope for further research on the various topics mentioned.

Query expansion is an important part of retrieval process and work needs to be done on improving efficiency by improving each mechanism and also try to combine different methods in order to maximize the effects.

While the methodologies concentrate on the results of their collection, there is lack of micro-evaluation of the tests performed. Such tasks are costly and onerous but provide qualitative clues for further explanation of the behavior of the method of query expansion that is studied. Thus far research on query expansion has not yet identified optimal levels for neither automatic query expansion nor interactive query expansion.

In ranking algorithms and other user centered evaluations, the user preferences have to be looked upon well. More research is upon developing automatic query expansion.

Future work can focus on reduced weighting for document expansion terms. Currently terms are added without their weight being diminished, unlike for the conventional approach, where expansion term weights are downgraded by two thirds.

References

[1]. Gerard Salton and Christopher Buckley. “Term – weighting approaches in automatic text retrieval”

[2]. Bodo Billerbeck and Justin Zobel. “Efficient query expansion with auxiliary data structures”

[3]. Bodo Billerbeck, Falk Scholer, Williams, and Zobel. “Query expansion using associated queries”

[4]. Efthimis N. Efthimiadis. “A user-centred evaluation of ranking algorithms”

[5]. Gerard Salton and Chris Buckley. “Improving retrieval performance by relevance feedback”

[6]. Giambattista Amari, Claudio Carpineto, and Giovanni Romano. “Query difficulty, robusteness and selective application of query expansion”

[7]. Bodo Billerbeck and Justin Zobel. “Techniques for efficient query expansion”

[8]. Mandar Mitra, Amit Singhal, and Chris Buckley. “Improving automatic query expansion”

[9]. Query expansion. http://www.faculty.washington.edu/efthimis

[10]. Shu Huang, Qiankun Zhao, Prasenjit Mitra, and Lee Giles. "Query expansion using topic and location"

[11]. Kevin Collins-Thompson and Jamie Callan. "Query expansion using random walk models"

[12]. Ellen Voorhees. "Query expansion using lexical semantic relations"

[13]. Hang Cui, Ji-Rong Wen, Jian-Yun Nie, and Wei-Ying Ma. "Probabilistic query expansion using query logs"

[14]. A. Rungsawang , A. Tangpong , P. Laohawee , and T. Khampachua. " Novel query expansion technique using Apriori Algorithm"

VITA

GRADUATE COLLEGE

UNIVERSITY OF NEVADA, LAS VEGAS

ABHISHEK BIRUDURAJU

Degrees:

Bachelor of Technology in Computer Science and Engineering, 2008

Jawaharlal Nehru Technological University, India

Thesis title: Consolidated study on Query Expansion

Thesis Examination Committee:

Chairperson, Dr. Kazem Taghva, Ph.D.

Committee Member, Dr. Ajoy K. Datta, Ph.D.

Committee Member, Dr. Laxmi P. Gewali, Ph.D.

Graduate College Representative, Dr. Muthukumar Venkatesan, Ph.D.