ANTICIPATING SEARCH NEEDS IN REAL TIME HELP DESK ENVIRONMENTS

MUJTABA HUSSAIN

Supervisors: ANDREW TURPIN AND FALK SCHOLER

Honours Thesis

School of Computer Science and Information Technology

RMIT University

Melbourne, AUSTRALIA

June 3, 2008

Abstract

While searching for and retrieving documents from a collection is a widely studied problem, most analysis assume that a user has time to formulate and enter a query in the search system. In many practical scenarios, such as a call center, the user is under pressure to find information for a customer. This research investigates techniques in which the spoken queries of a customer can be used to automatically retrieve relevant documents for the user by manipulating the characteristics of the query, especially the content.

We use the complete transcribed text of the spoken conversation between the user and the customer, and study various methods of generating queries suitable for input to a standard search engine.

We evaluate our techniques using the data from two real world call centerer environments, the Royal Melbourne Institute of Technology (RMIT) Computer Science and Information Technology Duty Programmer’s desk, and the call center of Medibank Private Limited. Our results show that there is decided advantage in manipulating the number of terms in a query to improve effectiveness; which for the purposes of this research is being measured using mean reciprocal rank of the retrieved resources.

Contents

1 Introduction

A help desk within an organization is a place where customers call when they have problems with the services provided by the organization. There are usually two or more people staffing the desk, who have intimate knowledge of the workings of the company and its services; these people are called help desk attendants. They handle the calls made by the customers, and subsequently try to resolve their problems.

Companies typically have support software installed which the attendants use to assist them in solving the problems presented by the customers. The support software usually involves interaction with a resource database which contains information on the company's processes, policies, procedures and products. The process of problem resolution involves attendants taking call from a customer, listening to the description of their problem, and then using the support software to interact with the database and retrieve resources that would help the attendants in solving the presented problem. The process usually involves the attendant manually entering a search query.

The process of obtaining the terms of the query from its spoken equivalent requires the attendant, through experience or some written protocols, to sift through the customer's spoken problem description and select the specific terms that would facilitate searching the database for the relevant resource. The process of sifting through what was said, obtaining specific relevant query words and then entering them into the system is time consuming and if this was automated, the whole process could be made faster.

This research begins from the assumption that the speech between the attendant and the customer has been converted to text in full, without any errors during the transcription. Such a transcript will in general be noisy and often contain poorly structured and ungrammatical structures. The query needs to be pruned to a more precise and relevant form before it becomes useful for an automated database search. The attendant, as mentioned previously, does this either on basis of experience or on the basis of some protocols.

This project will investigate the development of such a system. We have collected data from varied sources such as the Duty Programmers desk in the CSIT school at RMIT, and also the help desk at Medibank, a large Australian Health Insurance Company. This data is in the form of text queries and for each, the corresponding document that provides relevant information for that query are identified. The research will utilize these transcripts in order to develop algorithms to successfully prune large query texts into simple, small and more accurate sets of terms for successful search. It is these algorithms that will be the main result of this honours research.

All of us at one time or other have had to call a help desk or a call center where we have had to wait while the attendant attempts to find the proper resource to assist us. Successfully completing this research will improve the quality of service across call centerers of many companies and also improve the time taken to serve customers, particularly in emergencies (For example, an ambulance attendant calling a hospital to ask for medical background of a crash victim).

1.1 Research Questions

The primary aim of this research is to provide support for attendants in a call center or a help desk environment. This is done by a series of experiments with the aim of optimizing the query entered by the attendant into the search system. The varied sets of experiments performed by the researchers would attempt to answer the following questions:

1. Does the removal of noise terms (For example, “umms” and “aaahs” improve the effectiveness of search?

2. Does removal of words from a query which occur frequently in the database improve effectiveness of search? If so, what is the threshold of frequency where improvements are best?

3. Does specific weighting of terms in a query improve ranking of results?

2 Background and Related Work

In this section, we survey related work on spoken term detection and query retrieval from transcribed text. This has been divided into three sections, representing the three aspects of this research.

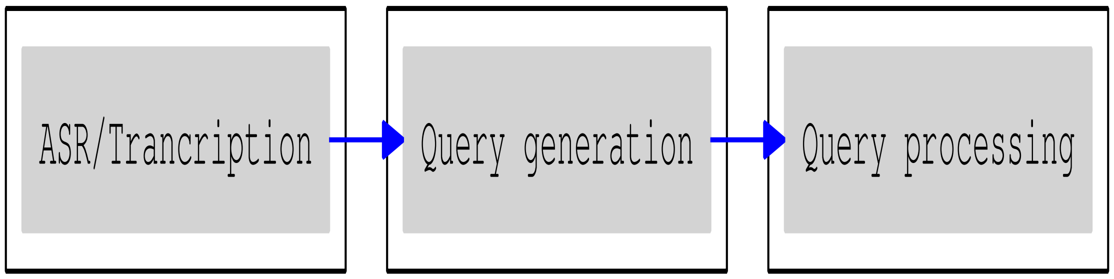

Figure 1: The three steps involved in this research

2.1 Automatic Speech Recognition and Transcription

2.1.1 Vocabulary Independent Spoken Term Detection

Mamou et al. (2007) identify two main problems with transcribing queries from spoken recordings. First, the transcription of the recordings has to be accurate for the retrieval system to return meaningful documents; and second the system needs to handle noise terms such as “umm” and “aaahs”.

Because of these possible restrictions an Automatic Speech Recognition (ASR) system cannot be relied upon to transcribe spoken text correctly and effectively as it is not possible to remove attenuation from transcribed text or to transcribe the complete text without errors. Instead of transcribing the text, Mamou et al. (2007) proposed converting the text into a phonetic series. This would represent the query as a series of phonetic transcripts or *phones*. Even though this process of phonetic transcriptions reduces error rates (of transcription) compared to standard transcription, this process still has difficulties in properly converting Out of Vocabulary (OOV) terms, something which might not happen in a word-for-word transcription using an ideal system.

Since both these approaches did not work individually, Mamou et al. (2007) propose a hybrid system which is capable of :

- recognizing OOV and in vocabulary(IV) terms;
- indexing both word and phonetic transcription; and
- merging all the information from both indexes.

The generation of an individual phone in this particular hybrid is based on the time duration between each phone. For example for the phrase **prosody research**, the OOV term **prosody** is converted into the subsequent series of phones : *p,r,aa,z,ih,d,iy* and each is indexed. A posting list of the phones is generated and phones with the time gap of less than 0.2 seconds get their posting lists merged. Using this technique, the term **prosody** can be generated in text

Mamou et al. (2007) advise that in any transcript, there are three errors that can emerge. They are: insertion of non-existing terms; deletion of existing terms; and, substitution of existing terms with non-existing ones. These errors collectively give rise to the Word Error Rate (WER) which is a metric used by the authors to measure the accuracy of the transcript.

Since conversion of spoken dialogue to a textual transcript requires a robust conversion system, which is an issue in its own right, this research shall be proceeding with the assumption that the spoken dialogue has been converted into textual transcript as accurately as possible. We do not consider the problem of transcription errors further in this thesis.

2.1.2 Spoken Document Retrieval

A vast resource of audio in the form of a dialogue comes from the radio. Johnson et al. (1998) used the audio tracks from the American Broadcast Radio and TV news programs as their input. Using an ASR called HTK (Hidden Markov Model Toolkit), they converted these duologue's into word transcripts. The particular ASR in question transcribed the dialogue into text by going over the said text twice to remove any discrepancies.

Using entire episodes of news shows and radio programs, the HTK system generated text that was then processed by the Okapi-BM25 Engine. This engine removed words deemed trivial by the internal protocol of the engine (stopping) and then removed common suffixes from words (stemming). In doing so, this engine also removed the inherent *umm's* and *aaah's* from the text as it deemed them to be stop words. This stopping and stemming increased the precision of this system by 0.7% compared to previous implementations, which had kept these OOV terms inherently.

Johnson et al. (1998) also used a word mapping technique whereby words like *Chechnia* (which is an OOV) was mapped to its proper spelling *Chechnya*. This mapping again increased the precision, this time by 2.0%. However the authors note that these results have emerged after testing the process on a subset of the transcripts of the audio originally obtained, and hence the figures indicating increase in precision cannot be completely relied upon. To a get a thorough understanding of the effect of the above mentioned procedures on precision, the authors recommend testing it against a larger database or collection.

2.1.3 Experiments in Spoken Document Retrieval

An interesting problem was observed by Nowell (1998) in the Okapi engine used previously by Johnson et al. (1998). Abbreviations under the Okapi engine ran the risk of being converted into stop words after conversion, and removed. For example, if the system came across *U.S*, the Okapi pre-processor converted the term to *us* and then the word would be stopped out of the final set.

This is a problem because abbreviations have the potential to be highly useful in distinguishing documents. A general workaround based on this observation that was made was to write abbreviations like the example given previously as *UxxxSxxx*. Once implemented, a general increase in precision of the documents was observed.

The whole concept of removing OOV noise words from queries does not extend to this thesis as we find out in experiments later on that, since these words do not occur in the document collection and are therefore indexed at all, removing them from the queries has no effect, which is the approach this research will be taking.

2.1.4 Automatic Call Segmentation

Instead of taking the entire transcript from the caller and the attendant, Park (2007) decided to split the call transcripts into several sections. The target data comprises of spontaneous speech between the customer and the caller and service representatives, otherwise known as attendant. The target call, based on the contents was then divided into 7 different sections. These sections were *"Greeting"*, *"Question"*, *"Refine"*, *"Research"*, *"Resolution"*, *"Closing"* and *"Out-of-topic"*.

These sections were the most generalized form of sections and indeed would vary across companies and transcripts, and sometimes even attendants would be a factor in the calculation of the appropriate sections.

Park (2007) used an off-the-shelf Automatic Speech Recognition (ASR) system to transcribe the spoken dialogue and they used a Support Vector Machine (SVM) to convert the transcribed dialogue into the relevant sections. Using all this, Park (2007) experimented on about a hundred automatically transcribed calls.

The off-the-shelf ASR gave Park (2007) a Word Error Rate (WER) of 42%. They subsequently trained the system on about 2,236 manual transcripts, which contained accurate utterance changes and speaker turns. Based on the training set, the system utilized a set of keywords to sift duologue's into transcripts. This was effective as the there are only two speaker types - customer and attendant and each of them almost always uses a distinct set of keywords. Park (2007) gives examples of keywords like *appreciate*(customer), *hold*(attendant). Using a set of probabilistic measures and the above mentioned set of keywords, the system classifies the attendant and the customer dialogue segments. This system, was evaluated with 2000 manual transcripts and showed a 74.67% F-score (which is defined as weighted harmonic mean of precision and recall) on average which is a decided improvement.

In addition to the above mentioned attendant/customer classification, the system also uses features like speaker identification, and positions of utterance in the transcript, to classify the spoken dialogue into one of the seven sections. The utterances were merged in the following two ways: adjacent utterances from the same section type were merged; then, small call sections between two longer call sections were merged by using the method of ω words per minute process of finding length of the section. Park (2007) experimented on about 100 randomly selected transcripts from a call center of an Information Technology company, which was made up of 13.2 hours of conversation. Using 10-fold cross validation, the system showed about 87.2% classification accuracy on average. This SVM based system, even on this small data set, showed an improvement on the previous approaches such as *Resolution only* method which had 75.4%, occurences and the *Maximum entropy based segmentation* which had 81.2% accuracy.

This segmentation could provide call centers with more detailed knowledge on the calls, thus enabling many of the applications used by them to search and sift more efficiently through the data. This segmentation however still uses an ASR, which according to Park (2007) was highly inaccurate in generating transcripts (42% WER) and could hence potentially classify the sections of transcripts incorrectly. In the case of this particular research, our aim is to find the most relevant documents from the collection, but we have not trialled the automatic segmentation method. Rather, we manually extracted the “Question” section. The idea of segmentation could be utilized as a future extension of this particular research as it provides ample opportunities for further classification of transcribed text.

2.1.5 Automatic Analysis of Call Center Conversations

In order to assist the call-center agent so that they could do their job better, and to assist the agency itself in collecting the statistics, Mishne et al. (2005) decided to evaluate the transcript of the spoken dialogue with respect to specific domains and then assign significance to certain sections of the transcript.

The assignment of significance to transcript sections is done in two steps. First, the pre-computation of word values takes place off-line. This is done by comparing the frequency of the word in a domain specific corpus to its frequency in an open-domain corpus. Now that the words have been assigned their domain specific importance, the values are normalized, with 1 representing domain association and 0 representing little association. Mishne et al. (2005) then assign fragment level importance by simply using the significance values of the words in that fragment. The values for these words were generated in the initial pre-computation.

Mishne et al. (2005) used about 2276 calls that were transcribed by an IBM research prototype transcription server which had a WER of 30%-40%. The Juru IR engine was used to index the corpus, and then the corpus was tested against queries. The authors concluded that they had been successful in segmenting the parts of the transcripts based on their significance, but because of the high WER and also because the corpus they had initially tested it with was not robust, they recommended further evaluation.

This paper is relevant because the authors show that marking sections of transcripts significant is useful in allowing for better evaluation by someone like an attendant. But Mishne et al. (2005) initially did significance evaluation by comparing words with open-domain corpus which may or may not skew the results. In our work, we compare the sections of the transcripts with the collection from which the results need to be taken to extract the queries.

2.2 Transcription and Query Generation

2.2.1 Query Expansion using Associated Queries

The query typed in by the attendant is often not optimal for fetching the most appropriate document. Therefore, the query is often changed or added to in order to improve its retrieval performance. Fitzpatrick and Dent (1997) proposed a method of query expansion which works as follows. For a query that is to be expanded, they gather up-to three past queries that are very similar to the current one and obtain the top 100 ranked documents for those queries. These 100 documents are merged forming an affinity pool, from which candidate terms for expansion are selected by running the original query against this pool and then using a TF.IDF(Term Frequency,Inverse Document Frequency) term scoring algorithm for selection. Fitzpatrick and Dent (1997) observed an improvement of 15% in average precision.

Billerbeck et al. (2003), on the other hand, decided not to collect the additional terms to add from the affinity pool but from the past queries themselves. They found out which past queries matched the current one and then used terms from those queries to expand the current one.

Billerbeck et al. (2003) conclude that expanding TREC-10 queries using associations and then searching full text was more effective by 26%-29% than no expansion, and about 18%-20% better than an optimised conventional expansion approach. However, this approach to query expansion requires a large amount of preprocessing. This research will use TF.IDF to isolate possible significant terms in the transcribed conversations, but instead of query expansion we use it to prune noise terms from transcribed queries.

2.3 Processing Queries and Document Engine

2.3.1 Stemming and TF.IDF Ranking

Kantrowitz et al. (2000) studied the effects of stemming terms from the collection, especially when a TF.IDF measure is being used. The authors generated 17,782 stem-equivalence classes containing about 80,000 words. Stemmers that were used were the *Porter Stemmer* and *Damerau-Levenshtein Edit distance*.

The authors used *Edit Distance*, *Edit Distance with complex edits* and *Prefix Stemmer* as their three algorithms and they used the TREC-2 corpora for testing. After their experiments, Kantrowitz et al. (2000) concluded that improvements in stemming accuracy yielded corresponding improvements in retrieval precision, however this was the case only with short queries. There was an almost linear improvement in performance with increase in coverage, which suggested to Kantrowitz et al. (2000) that further improvements in TF.IDF ranking are possible if domain specific lexicons for stem-classes are introduced.

In this research, we use the stemming for both transcribed queries and for document indexing.

2.3.2 Probabilistic Analysis of Rocchio Algorithm with TF.IDF for Text Categorization

One of the most well-known algorithms for text categorization is the *Rocchio relevance feedback* algorithm [Rocchio (1971)] and the major heuristic that this algorithm uses is the TF.IDF (term frequency/inverse document frequency) word weighting scheme. Joachims (1997) developed a probabilistic analysis of a TF.IDF classifier which the authors called **PrTFIDF**.

After experimenting with *Newsgroup* and *Reuters* data, Joachims (1997) concluded that PrTFIDF showed a performance improvement of about 40% in reducing error rates, showing that the probabilistic model is preferable to the heuristic TF.IDF method. But they also state that the probabilistic model is preferable only in a theoretical sense simply because in generating this probabilistic model certain assumptions were made and it was a combination of these assumptions that provided this model with its results. Because these assumptions are theoretical in nature, the authors of this research opt for TF.IDF and not the probabilistic model.

This research shows that TF.IDF has been used previously in IR, but not for improving effectiveness, as we aim to do.

2.4 Explanation of Terms Used

During the course of this paper, there are few terms used that require explanation. There explanation is as follows.

2.4.1 Inverted Index

Zobel and Moffat (2006) have shown that the inverted index is an efficient data structure for text query evaluation. An inverted index basically constitutes of a collection of unique terms of the collection called the *vocab* and an *inverted list*.

An inverted list consists of a series of postings matching each unique term in the collection. These postings are in the form of $f_t < d_1[f_{d_1,t}], \ldots >$, where f_t is the frequency of the term in the collection; d_i denotes that the term occurs in document i; and $f_{d_i,t}$ denotes the frequency of term t in document i.

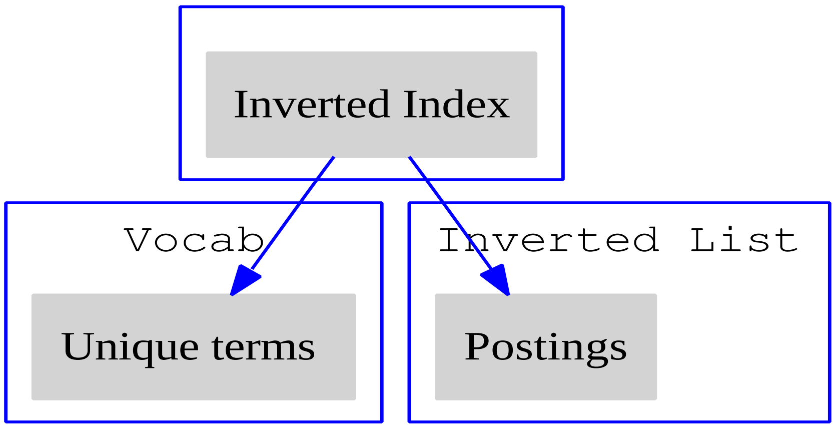

Figure 2: A basic structure of an Inverted File

2.4.2 T-test

When standard deviation of a population is unknown and has to be calculated from the data, doubts as to the statistical significance of the results arise. To ensure the integrity of the results and also to prove their statistical significance, a statistical test called the *Student's t-test* is performed.

A t-test can be described as any statistical hypothetical test performed to ensure that the results were achieved due to the performance of the algorithm and not due to chance. The metric used by the t-test is called the *p-value*. If the value of this metric lies under a pre-determined threshold (usually 0.05), then the test and thereby the results, are statistically significant.

For the purposes of this research, we used an **Independent two sample T-test**. We used this particular equation as our two data sets were equal in size. The said equation can be represented by:

$$t = \frac{\overline{X_1} - \overline{X_2}}{s_{\overline{X_1} - \overline{X_2}}} \ : \ s_{\overline{X_1} - \overline{X_2}} = \sqrt{\frac{s_1^2 + s_2^2}{n}}$$

s denotes the pooled standard deviation, 1 is group 1, 2 is group 2, and the denominator denotes the standard error of the difference between the two means.

2.4.3 Okapi BM25

Sparck-Jones et al. (2000) proposed the BM25 ranking function to rank documents based on their similarity to the given query. Okapi was the first search engine to use this function.

BM25, in and of itself, is a whole set of functions rather than a single function. It ignores the proximity of the query words completely and acts as a bag of words function, matching documents to query terms. One of the most widely-used instantiations of the function is:

$$score(D,Q) = \sum_{i=1}^{n}((IDF(q_i)) \times \frac{f(q_i,D) \times (k_1 + 1)}{f(q_i,D) + k_1 \times (1 - b + b \times \frac{|D|}{ADL}}))$$

Term	Explanation
$f(q_i, D)$	q_i's frequency in document D
$\|D\|$	length of the document
ADL	average document length within the collection
k_1	Free Parameter (usually 2.0)
b	Free Parameters (usually 0.75)
$IDF(q_i)$	IDF weight of the q_i

BM25, and some of its newer variants such as the BM25F (Which allows anchor text and document structure to be taken into account) are now part and parcel of Web search and represent state of the art search technology.

2.4.4 Mean Reciprocal Rank(MRR)

Moffat et al. (1999) define MRR as the average (over a set of queries) of the reciprocal of the rank of the first relevant document in a list of retrieved resources.

A search engine, given a query returns a ranked list of resources. The rank of each resource is an indication of how relevant the resource is to the query according to the search engine. We invert the rank of the resources to form a metric called *reciprocal rank*. For example, if query 1 gives a reciprocal rank of 0.3, query 2 gives a reciprocal rank of 0.5. To implement this metric across many queries or even huge sets of queries, we perform the mathematic function of *mean* across the reciprocal ranks to form *Mean Reciprocal Rank*. For example, two queries with reciprocal ranks of 0.3 and 0.5, the mean reciprocal rank is $\frac{0.3+0.5}{2}$. In our research, we are going to evaluate many different algorithms on a large number of queries and we use MRR as a metric to demonstrate which algorithm perofmed better than the rest.

3 Methodology

3.1 Data Collection

Two types of data are required for our experiments. The first, a complete collection that includes all the documents and the manuals that a help desk attendant or a call center attendant would use in order to solve a problem. Second, we require transcripts of conversations between the caller and the attendant. The former is the resource from which the solution should emerge and the latter is the one which will determine which solution is appropriate.

3.1.1 Duty Programmers Desk

The first source of data for this research was from the Duty Programmers Desk at RMIT. The initial collection that was obtained from the help desk was the complete wiki collection used as a pseudo-manual by the attendants at the Duty Programmers Desk [Programmers (2006)]. This wiki exists as a collection of text documents on of the internal servers of RMIT. The transcripts of conversations needed to be collected first hand as the DP Desk in RMIT does not record conversations. Ethics permission was sought and obtained from *RMIT Human Research Ethics* committee to record help desk conversations. The transcripts were obtained by recording the live conversations between students/staff coming to the desk asking for help and the attendant; then were recorded only after signing an informed consent form. These conversations were then transcribed verbatim, word for word, enunciation for enunciation and these became the queries for that particular collection. Noise and other random sounds that constituted the non-word set were transcribed literally upon the discretion of the researcher.

The main idea was to collect conversations containing OOV terms and random attenuation as one would expect in a spoken conversation over a telephone or face to face at any help desk or call center.

3.1.2 Medibank

The second set of data was collected from Medibank Inc. This text collection is a large corpus of manuals and documents that a Medibank call center attendant would use to solve a query, including, for example, health policies, and information on benefits. Queries were generated by selecting documents at random and manually deriving possible information for which that document would be an answer.

These queries were modeled very closely on the transcribed queries generated by recording the conversation at the RMIT Duty programmers desk. This was done in an effort to ensure that these artificially generated queries resembled the real world as closely as possible.

3.2 Software

To conduct searches, we used the Zettair Search Engine [Zobel et al. (2008)]. Zettair is an open source tool that includes an implementation of the Okapi-BM25 [Robertson and Sparck-Jones (1976)] similarity measure. By default, Zettair does not implement stopping but stemming is implemented, and we adhered to said parameters.

3.3 Evaluation Metric

We use Mean Reciprocal Rank (MRR) to evaluate search performance. This metric was chosen because in a call-center or a help desk environment, there is usually a document that will satisfy the query completely. MRR has been successfully used as a metric during the evaluation of the Question-Answering Track during TREC-8 [Vorhees (2000)] and also during TREC-11 [Crasswell et al. (2004)] and that was the added justification for using the specific metric.

3.4 Pruning Queries

This research aims to prune a query to a form which would retrieve the most relevant resources. We use the IDF value of terms in the query to select which terms should be discarded. This research assumes that the less frequently a word occurs in a collection, the more useful it is in identifying the document it is associated with. That is, a lower $\sum_{d \in D}(f_{d,t})$ implies that the t is more important.

4 Experiments

4.1 Reduction of Query Length

This experiment was conducted to determine if the length of query, independent of all other factors, had any effect on the rank of the retrieved documents. A set of fifty queries were generated from the collection of transcribed conversations collected at the CS and IT Duty Programmers desk. The queries were then broken into three forms. For example the query *How do I get vlc working on Linux?*, was parsed into

- Sentence - *how do i get vlc working with linux rather than windows?*
- Phrase - *how vlc work linux*
- Keywords - *vlc linux*

The sentence form obviously implied that the initial query could only be stripped of the noise terms and all other words left intact. The phrase form implied that noise terms and also common terms like ’I’ or ’on’ had to be removed. The final form had everything but the most essential terms removed from the query, where essential was determined by the researcher.

These sets of queries were run against the wiki index, giving the results in Figure 3.

It can be seen from the graph that the phrase and keyword type query has a better MRR than the complete sentence. The researchers determine it is so because in the sentence form, the search engine returns documents based on all the query terms, some of which might not concern the relevant topic, whereas in the phrase and keyword form, the offending query terms did not exist and the search results were much more focused. This ensures a better mean reciprocal rank. For example in the above mentioned example, the term “windows” misleads the search engine into retrieving documents that might not have anything to do with the main aim of the query.

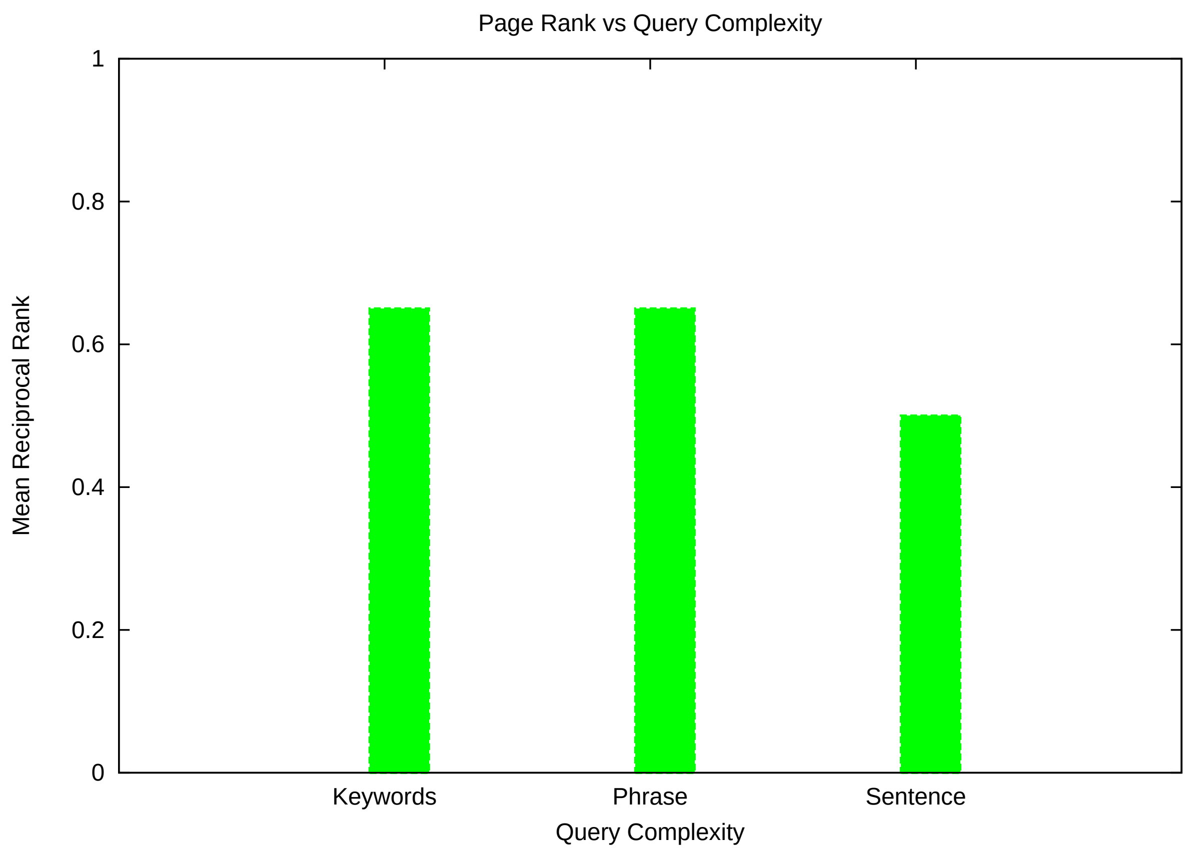

Figure 3: Reduction of Query Length

4.1.1 Results

The longer a query is, the more terms it has and hence more chance that less relevant documents will occur in the list of retrieved documents, which in turn might prevent the attendant from solving the problem quickly.

Upon experimenting with the collection by querying it with queries of varying lengths, the researchers determined that query length did have an effect on the ranking of retrieved documents. This was because reduction in length reduced the number of terms in the query that could bring up documents that did not specifically have anything to do with the topic in hand. Thus, in this particular experiment, phrase and keyword showed an improvement of 14% which is a significant improvement.

4.2 Query Pruning Based on IDF

This experiment was based on the idea that the importance of a term was determined by how many documents it occurred in. The terms in question are the terms of the query (t_q) and their frequency is calculated in the form of IDF obtained from the collection itself.

There are several intervening steps before the query can be run against the collection.

4.2.1 Parse Query

The query must go through the same pre-processing that the collection underwent and hence must be stemmed and stopped in an identical manner. This reduces the spoken query to a collection of words. For example, a sample query like *"What err, must a naturopathy provider umm do in order to pay benefits?"* becomes *"what must naturopathy provid do order pay benefit?"*. This is achieved by passing the query through Zettair's indexer.

4.2.2 Index Statistics

The IDF statistics are collected using a Zettair binary called **zet_cat** which returns the document frequency (t_f) for each term amongst other things. The following is a sample table based on a few terms of the query mentioned above, after it is parsed and its IDF has been extracted from the index.

t_q	t_f
naturopathy	60
order	143
what	187
provider	271

Once the statistics have been generated, the terms in the query are then arranged in ascending order of their t_f and that collection of terms then serves as a query. After one run, the first term, which has the lowest IDF, is removed and the query is run against the collection again. This process is continued till there are no more terms left in the query.

The aim of this experiment is to observe whether the removal of low IDF terms from the query improves the reciprocal rank. This experiment was run over a set of 150 queries from 30 different documents.

Figure 4 shows that pruning 60 percent of these queries gives the maximum mean reciprocal rank for the query. This result is backed up by statistically analysis (**t-test**, p-value of less than 0.05).

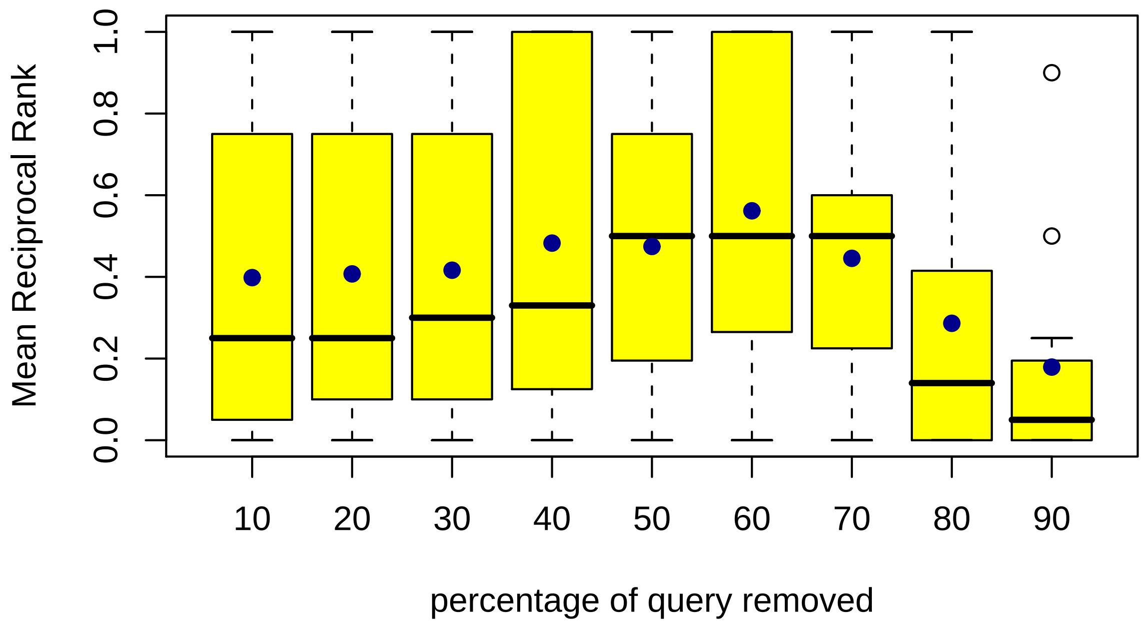

Figure 4: The median of the mean reciprocal values along with the standard deviation values. The black dots represent the mean values and the horizontal bars represent the median values.

4.2.3 Results

The idea behind this experiment was that not every word in the query is as important as that potential keyword which retrieves the relevant document. The aim of this experiment was to find those keywords and reduce the query as much as possible while keeping the most important keywords.

The experiments determined that IDF is an important factor in sifting relevant query terms from non relevant ones. Reducing the query one term at a time shows that at 60±5%, the query contains enough keywords to ensure the maximum mean reciprocal rank. This is because reduction in query terms with a low IDF ensures that the query contains only those words which are very important to the topic of the query. These words are very specific to the documents they relate thereby increasing the MRR. But if we throw away too many terms, chances are that we might discard important terms which might reduce the MRR.

4.3 Query Term Weighting

This experiment was conducted to determine whether giving a higher weight to certain terms in a query had any effect on the associated mean reciprocal rank. To put it into perspective of the entire research, the results would then be compared with the mean reciprocal obtained by pruning the query up to the optimum percentage obtained in the previous experiment which is 60±5%.

We experiment with increasing the weight of those query terms that are also title terms of some document in the collection. If a query term exists in the title of any document, then it is repeated in the query in order to increase its relative importance with respect to the rest of the terms in the query.

To ensure that a broad scale of weighting was achieved, the terms were incrementally repeated from 2 to 10 and were compared with the optimum pruning measure. The experiment was run over 150 distinct queries taken from 30 documents taken from the Medibank collection. The results are shown in Figure 5.

Figure 5 clearly shows that giving extra weight to terms which occur in the title index increases the mean reciprocal rank compared with its optimum pruned counterpart. However, repeating the title terms more than once offers no further benefits. Upon closed inspection, it is observed that it is the presence of title terms in the query which allow Zettair to bring up the relevant documents precisely whereas pruning the query at times removes the title term and hence does not perform as well.

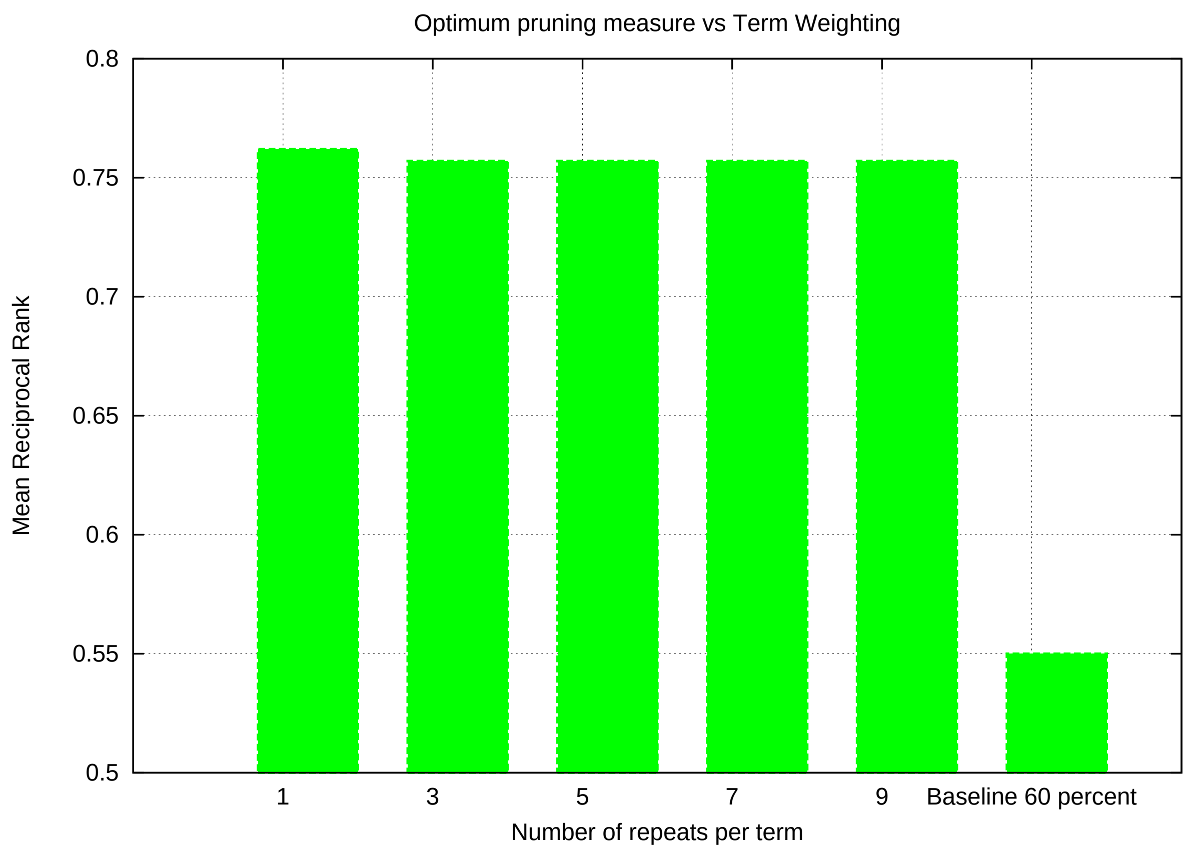

Figure 5: Giving higher weight to title terms

4.3.1 Results

The aim of this experiment was to see the importance that title terms have in a search. The process of giving more importance to words that occurred in the title of a document by repeating them several times was used to investigate this idea.

The results showed that there is a increase in performance when we repeat the terms more than once, but that further repetition has no further advantage. Upon comparison with the result of the query pruned to its optimum percentage (as shown in the previous experiment), the mean reciprocal rank was higher. This is due to the fact that increasing the weight of title terms improves the performance of the ranking function.

4.4 Pruning Query Based on Title Terms

The last experiment sought to exploit the features of the previous two. In this experiment, we parsed the query through Zettair and then arranged the terms in the query based on two algorithms which are implemented sequentially. They are:

- separate the set of query terms into groups, those that occur in titles and those that don't; and
- order the terms in each group by ascending order of their IDF.

Once the queries were arranged in that order, they were initially run against Zettair intact. After each run, terms were removed from the query . First, terms were removed from the non-title word group and once this group was depleted, terms were removed from the title-group. Figure 6 shows the MRR for different proportions of removed words.

4.4.1 Results

The aim behind this experiment was to combine the two approaches into a hybrid scheme containing the positive approaches of the previous algorithms. The results showed that if the query does not contain any title terms, then pruning it to about 60% improves the mean reciprocal rank compared to using the query in its full form. But if the query contains title terms, then pruning has no further benefits. This is so because when the query contains title terms, the search engine in question is extremely good at finding the documents whether or not the query is pruned or not.

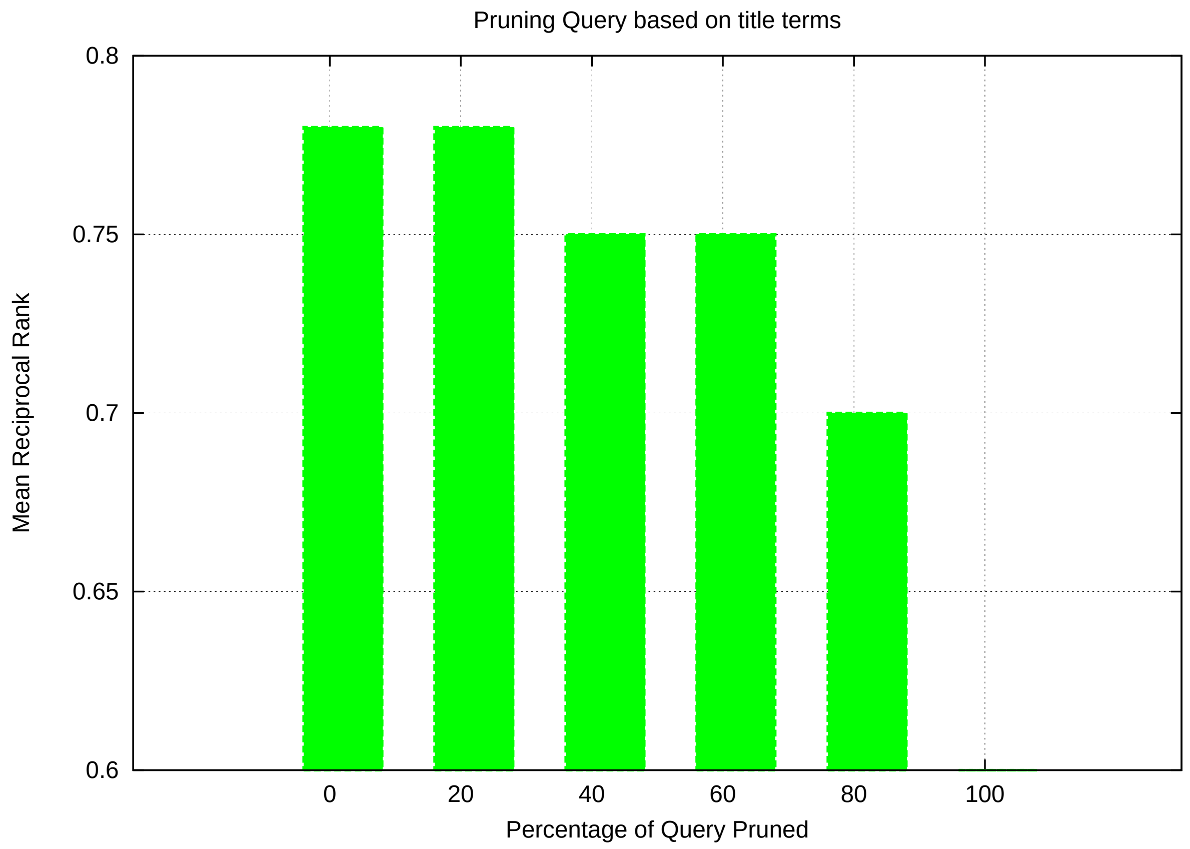

Figure 6: Hybrid Approach

5 Conclusion

In help-desk environments, there is a need to answer a customers query in the least amount of time as possible. In this research we consider the problem of effectively retrieving relevant resources from a help-desk database based on spoken queries.

We initially evaluated the effects of query length on the mean reciprocal rank and found that decreasing the number of terms did affect the MRR. Hence we targeted the research at finding the degree of pruning to achieve the best result.

We rearranged the query terms based on their IDF and found that after pruning the query to 60$\pm$5%, the mean reciprocal rank was at its highest and that further pruning provided no further benefits. Statistical analysis demonstrated the validity of this result.

To examine whether further improvement could be achieved, selective weighting of query terms was implemented which was inherently biased towards terms that occurred in the titles within the documents. This procedure provided further improvement in the mean reciprocal rank.

As a result of these approaches, a hybrid approach was developed which arranged the query terms into two groups, one containing title terms and the rest formed the other group. Both were arranged in order of ascending IDF and then the query was pruned to he previously achieved benchmark of 60±5% and it was found that if the queries contained title terms, than pruning held no real advantage but if the query did not contain title terms, then pruning provided real benefits in improving search effectiveness.

The aim of this research was to improve effectiveness of service in call center and help desk environments. We have shown experimentally that our approach, especially the hybrid approach, improves the effectiveness of search involved. Hence we recommend this approach of combining IDF title and document terms and selective pruning as an improvement on simple ad hoc search in said environments.

6 Future Work

6.1 Automatic Speech Recognition

The researchers manually transcribed the recorded conversations and thereby achieving a WER of 0%. This is not possible when the data increases or when the query set increases. This research can be extended into the area of Automatic Speech Recognition. This would increase the WER and thereby reducing the accuracy of the transcripts and thereby the accuracy of the results. However, the researchers who would implement an ASR would have to decide on an accuracy/ease-of-transcription trade-off.

6.2 Larger Collections

The results of this research were obtained for two specific collections. The first collection involved insurance data from Medibank Private and the other involved technical data from the

CS IT Duty Programmers desk, RMIT. Consistency of the results can only be maintained if the above mentioned algorithms are evaluated across a broad range of data, involving a number of different collections.

6.3 Real Queries

The queries used in this research were either collected at the Duty Programmers desk by recording conversations between a customer and an attendant or were manually generated by the researcher. The former set of queries formed a small subset of the total queries used as collection of said conversations was hampered due circumstances outside the control of this research. The latter set of queries formed the larger percentage of the total set but were inherently not closely related to the real world queries as they had been generate by the researchers themselves.

To ensure that these results are scalable across collections and data, more real world queries need to be collected for the algorithms to evaluate. This can only be done if the queries themselves are collected from a large resource in addition to the data itself.

References

Billerbeck, B., Scholer, F., Williams, H., and Zobel, J. (2003). Query expansion using associated queries. RMIT University, Melbourne, ACM.

Crasswell, N., Hawking, D., Wilkinson, R., and Wu, M. (2004). Overview of the trec 2003 web track. CSIRO, National Institute of Science and Technology.

Fitzpatrick, L. and Dent, M. (1997). Automatic feedback using past queries: Social searching. In *SIGIR 97 Philadelphia PA, USA*. ACM.

Joachims, T. (1997). Probabilistic analysis of rocchio algorithm with tfidf for text categorization. University of Dortmund, Germany, ICML-97.

Johnson, Jourlin, Moore, Jones, and Woodland (1998). Spoken document retrieval for trec-7. In *Seventh Text Retrieval Conference*. Department of Computer Science and Engg, Cambridge, TREC-7.

Kantrowitz, M., Mohit, B., and Mittal, V. (2000). Stemming and its effects on tfidf ranking. Canegie Mellon University, ACM.

Mamou, Ramabhadran, and Siohan (2007). Vocabulary independent spoken term detection. IBM Haifa and IBM New York research labs, ACM.

Mishne, G., Carmel, D., Hoory, R., Roytman, A., and Soffer, A. (2005). Automatic analysis of call-center conversations. University of Amsterdam and IBM Research Lab, Haifa, ACM.

Moffat, A., Witten, I., and Bell, T. (1999). *Managing Gigabytes*. Morgan Kauffman, second edition.

Nowell (1998). Experiments in spoken document retrieval. dera-sru, TREC-7.

Park, Y. (2007). Automatic call section segmentation for contact-center calls. IBM TJ Watson Research Center,NY, ACM.

Programmers, D. (2006). Dp faq. https://inside.cs.rmit.edu.au/support/dp:home.

Robertson, S. and Sparck-Jones (1976). *Relevance Weighting of search terms*, pages 129 – 146. Journal of American Society for Information Science.

Rocchio, J. J. (1971). Relevance feedback in information retrieval. In *The Smart Retrieval System*. NIST, Prentice-Hall.

Sparck-Jones, Walker, S., and Robertson, S. E. (2000). A probabilistic model of information retrieval. University of Cambridge, Elsevier Science.

Vorhees, E. M. (2000). The trec-8 question answering track report. TREC, National Institute of Standards and Technology.

Zobel, J. and Moffat, A. (2006). Inverted files for text search engines. RMIT University and University of Melbourne, ACM.

Zobel, J., Williams, H., Scholer, F., Yiannis, J., Heinz, S., Lester, N., Webber, W., Moffat, A., and Vo, A. (2008). Zettair search engine. http://www.seg.rmit.edu.au/zettair/.