Design of Arabic Information Retrieval System based on Hybrid Stemming Approach

By

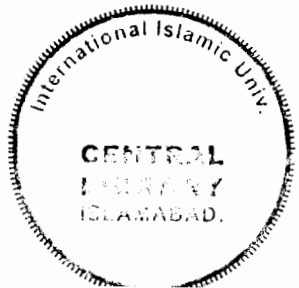
Bilal Khaliq (238-FAS/MSCS/F05)
Jehangir Rahman (251-FAS/MSCS/F05)

Supervised By

Dr. Syed Afaq Hussain

Department of Computer Science
Faculty of Basic and Applied Sciences
International Islamic University
Islamabad, Pakistan

(2007)



19-7-2010

MS
006.3
BID

1. Soft computing
2. Hybrid computers

EMD

DE
12/4/2012

MS

BJD

D.E.

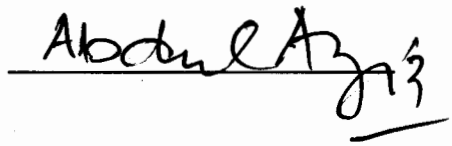
28-2-4

Final Approval

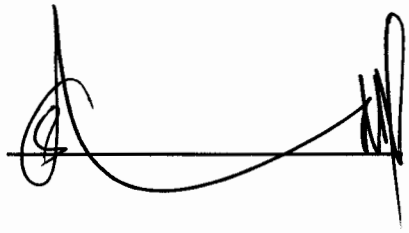
Date: _____

It is certified that we have read the thesis titled **"Design of Arabic Information Retrieval System based on Hybrid Stemming Approach"** submitted by Bilal Khaliq, Reg. No. 238-FAS/MSCS/F05 and Jehangir Rahman, Reg. No. 251-FAS/MSCS/F05 which in our judgment this thesis is of sufficient standard to warrant its acceptance by the International Islamic University, Islamabad for the award of MS in Computer Science degree.

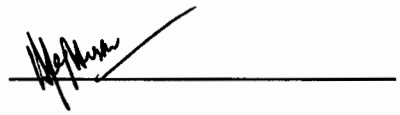
External Examiner
Dr. Abdul Aziz
Professor,
Department of Computer Science,
Riphah International University, Islamabad.



Internal Examiner
Mr. Shakeel Ahmad
Lecturer,
Department of Computer Science,
International Islamic University, Islamabad.



Supervisor
Dr. Syed Afaq Hussain
Professor,
Department of Computer Science,
Air University, Islamabad.



A dissertation submitted to the

Department of Computer Science
Faculty of Basic and Applied Sciences
International Islamic University, Islamabad
as a partial fulfillment of requirements for the award of
the degree of

MS in Computer Science

DEDICATED TO THE MARTYRS IN THE TRAJIC LAL MASJID INCIDENT, COINCIDING WITH OUR WORK, WHOSE SUFFERINGS LEFT A PROFOUND IMPACT ON OUR HEARTS

Declaration

We hereby declare that this work, neither as a whole nor as a part has been copied out from any source. It is further declared that we have developed the application and accompanied report entirely on the basis of our personal efforts and under the sincere guidance of our supervisor, Dr. Syed Afaq Hussain. If any part of this project is proved to be copied out from any source or found to be reproduction of some other project, we shall stand by the consequences. No portion of the work presented in this dissertation has been submitted in support of any application for any other degree or qualification of this or any other university or institute of learning.

Bilal Khaliq
(238-FAS/MSCS/F05)

Jehangir Rahman
(251-FAS/MSCS/F05)

Acknowledgement

All praise to Allah (SWT) who has guided us in undertaking work on the Arabic Language and has helped us through at each step when we felt there was no hope of pulling through.

We are thankful to our supervisors, Dr. Syed Afaq Hussain and Dr. Tauseef-ur-Rahman for their kind help and supervision. We are specially obliged to the latter for introducing us to the topic of Arabic Information Retrieval and going out of the way in providing guidance and assistance.

We would also like to thank our colleagues and the faculty members at the University for their help and support; with special thanks to our friend Muhammad Adeel for helping us in the coding and implementation of the Information Retrieval System.

Finally, we owe a lot to our beloved parents and our families for their love, guidance, moral and financial support.

Project in Brief

Project Title:	Design of Arabic Information Retrieval System based on Hybrid Stemming Approach
Undertaken By:	Bilal Khaliq Jehangir Rahman
Supervised By:	Dr. Syed Afaq Hussain
Start Date:	April, 2007
Completion Date:	August, 2007
Tools and Technologies:	MS Visual C++ 2005 Lemur Toolkit for Information Retrieval
Documentation Tools:	MS Word, MS Excel
Operating System:	MS Windows XP Professional
System Used:	Pentium 4, 1.3 GHz, 758 MB RAM

Abstract

Stemming is of critical importance for Arabic, which is a highly inflected Language, having a significant effect on Arabic document retrieval, as has been established by past research.

In this study we have experimented with a combined approach to stemming which we refer to as *Hybrid Stemming*. This approach merges the benefits of two stemming techniques, Light Stemming and N-Grams based conflation of words. Light Stemming nullifies the effect of some prefixes and suffixes that are appended to words, while N-Gram matching can conflate broken plurals diminishing the effect of infixes, and certain other affixes not handled by light stemming. Although hybrid stemming has been explored before using wordlists, it has so far not been implemented and evaluated in an Information Retrieval System (IRS) due to unavailability of a standard sizeable dataset in the past. We accomplished to establish the true standing of the Hybrid Stemming approach in practice by designing an IRS based on the standard TREC Arabic Dataset.

Our study revealed dismal prospects for the Hybrid Technique for Stemming. Past research gave motivating results for hybrid stemming without contrasting how strong a role the light stemming part plays in the combined approach. We showed that a good light stemmer which is able to conflate most word variants gives little opportunity to N-Grams matching procedure to improve the retrieval performance any further.

Table of Contents

Chapter # **Contents** **Page #**

1. Introduction

1. Introduction

No one doubts that the acquisition of valuable information is of primary importance for progress and development in our society. This information may not be obvious and usable unless properly extracted and presented in a manner which makes it valuable for people to use. Numerous fields are dedicated towards the effort to overcoming the problem of information overload. Information Retrieval (IR) is a science which helps to bridge the gap between humans and information systems by extracting the required information for the inquirer.

Information is not limited by the boundaries of a region or society. Just as it spans across cultures, societies, regions and languages, IR research also faces the challenge of overcoming the language barriers. English dominates as an electronic information resource and hence in the past has been the main focus of most of the IR research. But in recent years there has been growing amount of such resources in other languages like Chinese and Semitic languages like Arabic, and hence are beginning to receive more and more attention by researchers. We have chosen to investigate techniques for monolingual IR in Arabic as it is one of the most widely used languages with vast information starting to become available electronically. It is also one of the most challenging languages to process for IR.

Central to the problem of IR is the *vocabulary mismatch* problem (Larkey, 2006). The same information needs are expressed using different terms inflected differently in the query than in the relevant document. Stemming techniques are used to deal with the morphological variations in a particular language in order to *conflate* the different word forms to the same basic root or stem. Arabic is a highly inflected language having a complex morphology and hence requires more sophisticated analysis techniques as compared to English. Many of the approaches that are applicable to English are not directly applicable to Arabic. Rather these approaches need further enhancements or a completely different approach needs to be taken in order to overcome the complexities.

The literature contains many strategies that have been employed by researchers in this field including light stemming, lexicon-based, combinatorial and pattern based approaches. These techniques can be purely statistical but mostly they are linguistically oriented and sometimes a combination of the two. A comprehensive survey of Arabic Morphological and stemming techniques was done by Al-Sughaiyar et al.(2004).

A hybrid approach for stemming, combining a linguistic and statistical approach based on string similarity-measure, was explored in the literature (De Roeck, 2000; Mustafa, 2005). *Hybrid Stemming,* as we will refer to it, combines two techniques, affix removal or light stemming (linguistic) and N-gram based matching (statistical) reaping the benefits of the two approaches. The hybrid stemmer has been evaluated by researchers using words, not actual documents retrieval due to non existence of any standard sizeable dataset. We have furthered this earlier work by implementing the hybrid stemming approach in an information retrieval system using the standard Arabic TREC[1] dataset with evaluation measures provided by the TREC evaluation resources.

1.1 Information Retrieval (IR)

The goal of IR is to find documents *relevant* to an information need from a large document set. This information need is usually expressed by a query generated by the user. The query is some times referred by evaluators of IR systems as 'topics' since they are used to judge the accuracy of a pool of documents based on a certain topic query.

1.1.1 Monolingual vs. Cross-lingual Retrieval

At this point we distinguish between two kinds of retrieval method which come up in discussions about multilingual retrieval. Monolingual retrieval is one in which the query and the documents are in the same language whereas as in cross-lingual retrieval the query and the document are in different languages. Our concern is only with monolingual retrieval.

[1] Text REtrieval Conferences (TREC), http://trec.nist.gov

1.1.2 Vocabulary Mismatch Problem

A central problem in IR is the vocabulary mismatch problem as mentioned earlier. When we want to search for a word in the documents, say, 'swimming', we would also like that the search should retrieve documents which contain every other word form of the word 'swimming' like 'swimmer', 'swim' and 'swam'. This is what is referred to as the *vocabulary mismatch* problem. Word form analysis or *morphological analysis* would conflate these forms to a single form, the *lexeme,* for indexing. Instead of indexing all words, *conflation* of the words using morphological analysis techniques would index the lexeme 'swim' representing all the other forms, 'swimming', etc. The operations performed on the term to reduce it to a root or stem is known as *stemming.* For example stemming the word 'swimming' refers to the operation of stripping the suffix 'ing', removing the extra 'm', leaving behind the stem/root, 'swim'.

1.1.3 Workflow of an IR system

A simple workflow model for IR, highlighting the role of stemming, is shown in figure 1.1 below. After tokenizing the documents to extract all the terms in the document collection, stemming is applied to the extracted terms. The stems, instead of the term are used to create the inverted index file tabulating the stemmed terms and the corresponding document in which the terms occur. When a query is submitted, it too must be stemmed, resulting in the stemmed terms to be used for querying. Finally, using a matching procedure, the documents containing the queried termed are retrieved.

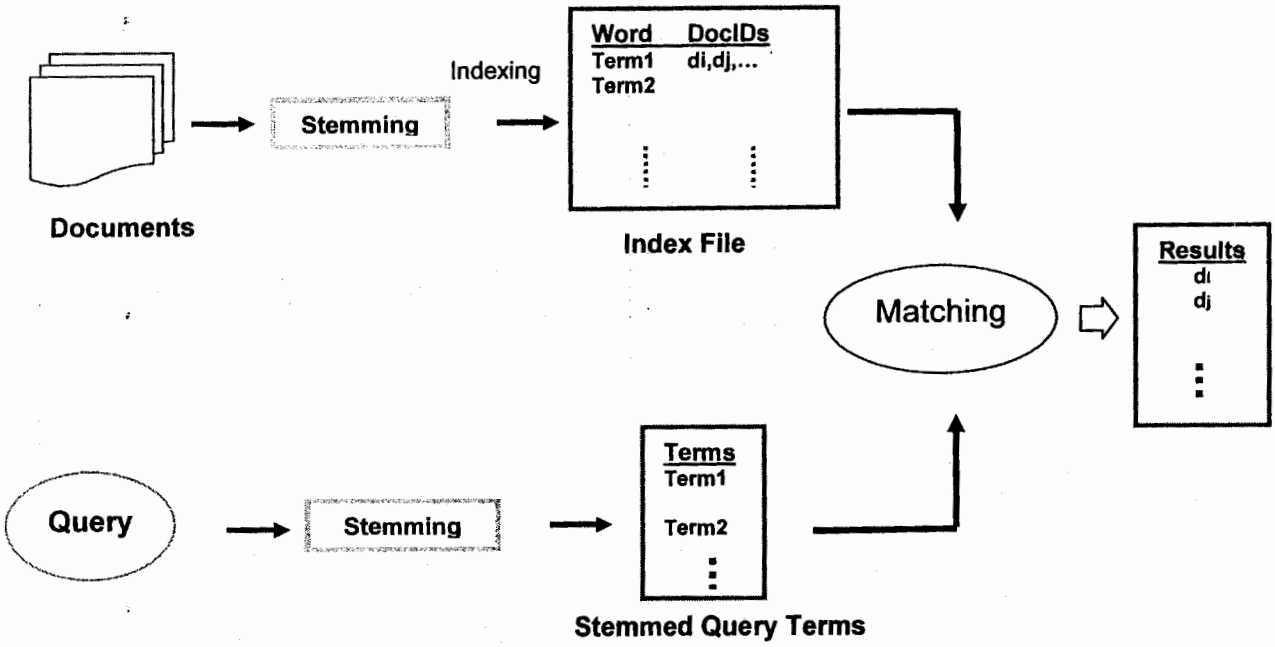


Figure 1.1: Workflow for IR highlighting role of stemming

1.1.4 Retrieval Models

The algorithm and procedures used in a system characterize each system and define its techniques, which can differentiate between an efficient and effective IR system, and one that fails to provide the users with satisfactory results. There are a number of information retrieval models that utilize these techniques in the retrieval process of which the three main categories are Boolean Model, Vector Model and the Probabilistic Model. These are briefly described by Abu Al-Khair (2003) as follows:

Boolean Model: This is a simple classic model for information retrieval. As the name implies, it uses the theory of Boolean algebraic logic. The query is expressed as a Boolean expression of term using "AND", "OR" or "NOT" operators. Documents are retrieved based on whether the term in the query is present in the document. The document is given a value or weight of 1 if the term is present in a document; if absent then a weight of 0 is given.

Vector Model: This is one of the most popular models used for retrieval. It is simple yet not as naïve as the Boolean model. In this model the degree of similarity between the documents in the collection and the query given by a user is evaluated using similarity coefficients such as the Dice coefficient or the Cosine coefficient. It uses non-binary term weighting with each term having a weight from 0 to 1. The matching process is based on these weights that are incorporated in the similarity measure to determine the correlation between a document and a query.

Probabilistic Model: The systems using this model attempt to estimate the probability that the user will find documents relevant to his query. It assumes that this probability depends on the query and the document representation or description, and there is a subset of documents that are relevant. All documents in this subset are predicted to be relevant and documents that are not in it are predicted to be non-relevant.

1.2 Arabic Language Processing

1.2.1 Characteristics of the Arabic Language

There are certain distinct characteristics of the Arabic language as explained by Abu El-Khair (2003) in his dissertation. Unlike English, Arabic language is that it is written from right to left. Arabic language consists of twenty eight characters that are consonants and vowels :{ ا ب ت ث ج ح خ د ذ ز س ش ص ض ط ظ ع غ ف ق ك ل م ن ه و ي }. Each character could have up to four different forms in writing according to the position of the letter in the word whether in the middle, beginning, end or isolated. In addition of the twenty eight characters three more symbols can be added to the character or the word and change the pronunciation of the character or the meaning of the word, which makes a total of thirty one characters. There is no concept of abbreviations and acronyms in Arabic. The plurals are almost always formed irregularly depending on the root and singular form of the word.

Certain vowels (represented as diacritic marks) may be included and or excluded from writing. There are eight main diacritic marks in Arabic: (ـِ ـُ ـَ ـّ ـْ ـٍ ـٌ ـً). These can change the meaning of a word in a sentence depending on the position of placement. For example, كَتَبَ: he wrote, كُتُب : books كُتِبَ: written.

The Arabic language belongs to a Semitic language group. These languages have a common grammatical system based on a root and pattern structure. Most of the Arabic words are morphologically derived from a short list of generative roots, which are bare verb forms. These roots are mostly trilateral, sometimes quadrilateral, and rarely pentaliteral. The root gives the basic lexical meaning of the word.

1.2.2 Arabic Morphology and Related Concepts

Stemming is of key importance for Arabic IR. In order to understanding stemming and its key role for Arabic, it is first important to get to known some Arabic morphology and other basic terms and concepts related to it. Various concepts and terminology have been used by researchers in the literature while discussing Arabic morphology. We have followed the terminology and concepts used by Al-Sughaiyer et. al. (2004):

i) Word: It is single isolated lexeme surrounded by two spaces that gives a meaning. For example, المحمدون is one word. Words in Arabic are variant form of some root. All words are classified as nouns, verbs or particles.

ii) Morpheme: It is the smallest element of a word imparting a function or meaning to the word. Some morphemes may themselves be words. For example, in English we have *dis-*, *-ment*, *agree*, similarly in Arabic we have ـون, أكل, etc.

iii) Roots: It is the original form of the word, providing the basic meaning, before being transformed into other variant forms. This is different than the *word base* or stem in

English, formed simply by stripping of affixes. For example in the word, المحمدون, the root is ح, م, د.

iv) Stem: It is a morpheme or set of morphemes which can be appended by an affix. Independently, it expresses a central idea or meaning. For example, in the word, المحمدون, the stem is محمد after the prefix and suffix.

v) Affixes: In Arabic affix can be of three types prefixes (attached in the beginning), suffixes (attached at the end) and infixes (attached in the middle). These morphemes are appended to stems or roots to form new words or meanings. E.g. Prefix: From كتاب, to الكتاب, by appending ال in the beginning; Suffix: From كتاب, to كتابها, by appending ها at the end; Infix: From كتاب, to كاتب, by appending ا in the middle.

vi) Broken Plurals: These are irregular plurals formed not following any pattern. Like in English, the irregular plural of *goose* is *geese.* In Arabic, as opposed to English which has very few irregular broken plurals, they are found extensively.

vii) Stemming: This involves conflation based on orthogonal similarity i.e. by dropping or nullifying the effect of affixes, roots or stems are left for conflation. In English, it is simply dropping of suffixes, -s, -ing etc which yields us the word base or stems. Stemming is more sophisticated and more important in Arabic because there are too many variants of a word in the text as compared to English. For example, the word, الوالدان, after applying stemming becomes والد.

viii) Mophological Analysis: This refers to a deep analysis of a word to break it up into its constituent part identifying the morphological pattern it follows and which root it belongs. For example for the word, الضاربون , the analysis would output the suffix, ون , prefix, ال , stem, ضارب, root, ض ر ب and pattern, فاعل .

ix) Conflation: This refers to grouping together of the words that belong to the same root or stem. Depending on the stemming approach used multiple words may be conflated to a single root or stem which is then used to represent all the words in performing retrieval. This is the case in for example the affix stripping approach. E.g. the word الكتاب and كتابها get conflated to the same class/group with the common stem كتاب after the removal of the prefix and suffix of the two words respectively.

In other cases the group of words forms a cluster with no single word representative in which case the whole class of words replaces each word in the class while performing the retrieval task. This is what we see when using N-Gram technique for stemming.

1.2.3 Conflation in Arabic

The need for conflation in Arabic can be seen from the example given in the table 1.1, below. It shows 76 word variants of the word *child*, (طفل). Although the number of prefixes and suffixes are small, the huge number of word variants occur due to the inclusion of conjunctions (ب, ل, و, etc) as part of the word, thus multiplying their total number. Hence as this example shows, it becomes imperative for Arabic IR applications to include conflation techniques in order to conflate these variants.

Arabic word	English translation	Arabic word	English translation	Arabic word	English translation	Arabic word	English translation
أطفال	children	اطفالهن	their children	بطفل	by child	فالطفلة	then the child
أطفالا	children	اطفالي	my children	بطفلة	by child	فطفل	then child
أطفالنا	our children	الأطفال	children	بطفلتنا	by our child	كأطفال	as children
أطفاله	and his children	الاطفال	children	بطفلته	by his child	كالطفل	as the child
أطفاله	his children	الطفل	the child	بطفله	by his child	لأطفال	to children
أطفالها	her children	الطفلان	the children	بطفلها	by her child	لطفلها	to her child
أطفالهم	their children	الطفلة	the child	بطفلهما	by their child	للطفلة	to the child
أطفالهن	their children	الطفلتان	the children	بطفلين	by children	وأطفالنا	and our children
أطفالي	my children	الطفلتين	the children	بطفليها	by her children	والأطفال	and the children
اطفال	children	الطفله	the child	طفل	child	وبطفل	and by child
اطفالا	children	الطفلين	the children	طفلا	child	وبطفلين	and by children
اطفالك	your children	بأطفال	by children	طفلان	children	وطفلة	and child
اطفالكم	your children	بأطفاله	by his children	طفلاها	her children	وطفلتان	and children
اطفالكن	your children	بأطفالها	by her children	طفلة	child	وطفلنا	and our child
اطفالنا	our children	بالأطفال	by the children	طفلت	child	وطفلها	and her child
اطفاله	his children	بالطفل	by the child	طفلتان	children	وطفليه	and his children
اطفالها	her children	بالطفلة	by the child	طفلتة	his child	وطفليها	and her children
اطفالهم	their children	بالطفلتين	by the children	طفلتنا	our child	ولأطفالها	and to her children
اطفالهما	their children	بالطفلين	by the children	طفلته	his child	وللطفل	and to the child

Table 1.1: Word Variants of *child* (طفل)[2]

Whether to conflate on root or stems is a debatable matter. Earlier it was thought that roots are better than stems for indexing. But these works were carried out using small, nonstandard datasets. In recent years with the availability of standard datasets, like the Text Retrieval Conferences (TREC) track, 2001, new evidence shows no definite differences between roots and stems for indexing (Larkey, 2006).

[2] Chen & Gey, 2002

1.3 Stemming Techniques

The techniques investigated for stemming and morphological analysis are numerous. We will overview the main course of research undertaken by researchers as seen in the literature which justify our research endeavor.

A number of classifications have been proposed for stemming techniques in the literature. Here we have chosen to broadly classify them as *linguistic* and *statistical* approaches with further classifications under this broad classification as found in the literature. This is shown in the figure 1.2, below:

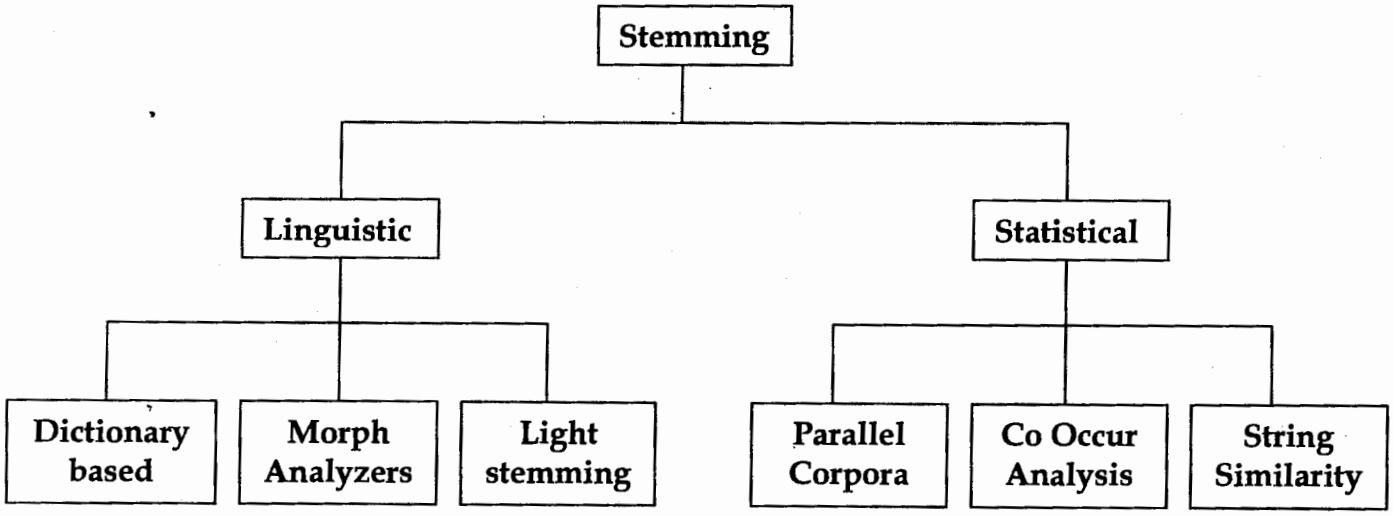


Figure 1.2: Classification of Stemming Techniques

1.3.1 Linguistic Approaches

These require expertise of the structure and rules of makeup of a language and understanding of the orthographic variation of words of a language. These techniques are the main focus of most researchers in Arabic involving careful processing of text for IR. Some of the avenues that are taken in the literature are briefly described below:

Dictionary Based: Also referred to in the literature as the *table lookup* approach, this early work on stemming uses manually constructed dictionaries. It involves building a large table storing Arabic words found in natural text with their corresponding Arabic parts (root, stem and affixation). As shown in table 1.2, multiple entries may exist for a word with the same spelling to cope with possibility of multiple analysis i.e. different affixations to the same stem/root.

Natural word	Stem	Root	Prefix	Suffix
AlvHlyl التحليل	vHlyl تحليل	Hll ح ل ل	Al ال	
...				
vHlyl تحليل	vHlyl تحليل	Hll ح ل ل		
...				
vHlylh تحليله	vHlyl تحليل	Hll ح ل ل		h ـه
...				
wAlvHlyl والتحليل	vHlyl تحليل	Hll ح ل ل	wAl و ال	
...				
wbvHlylAv وبتحليلات	vHlyl تحليل	Hll ح ل ل	wb وب	Av ات
...				

Table 1.2: Lookup Table showing multiple entries of same word [3]

Morphological Analyzers: A morphological analyzer is a tool for analyzing the morphology of an input word. The analyzer includes a recognition engine, identifying affixes and finding a stem or root within the input word. They output the morpho-syntactic information related to each word. That is they identify the affixes, the root, the stem, the pattern of the word. Some apply part of speech tags to the words. They usually use lexicons for affixes and stems/roots. For the purpose of IR not all information output by the analyzer is utilized; only the stem or root is used. Several morphological analyzers have been built for Arabic. But few are available publicly for evaluation in order to do comparison studies.

[3] Al-Sughaiyer et. al., 2004

Affix Removal (Light Stemming): This is a simple technique used for conflation. Also referred to in the literature as light stemming, it involves identification and stripping away of certain affixes which are carefully chosen and studied by linguists. In English language we mostly use light stemmers only for suffix removal whereas in Arabic it involves removal of suffix, prefix or sometimes even infixes in a words.

It is important to emphasize the effects of over stemming and under stemming in the case of light stemming which result in poor conflation. If too few affixes are chosen then the number of a classes that are formed for related words are numerous with related words getting grouped separately. On the other hand if too many affixes are chosen for stripping it results in fewer classes being formed with unrelated words being grouped together.

1.3.2 Statistical Approaches

These approaches are known to be language or domain independent involving little or no application of linguistics. As compared to linguistic approaches, there have been relatively few studies using statistical analysis.

Recently statistical stemming techniques are being explored which involve successor variety measures but not many studies exist as yet. Other statistical techniques use co occurrence analysis which are usually used to refine classes after application of stemming but can also be used to build stem class. The technique mainly used for statistical analysis is based on the conflation of words using *string similarity measures involving n-grams*.

Successor Variety: Successor Variety (SV) Stemmers are used to determine word morpheme boundaries based on the distribution of letters in a large body of text. The SV of a string or substring is the number (variety) of different characters that follow the string in words in the text. For the first letter substring of a word this number is high as there may be many words in the text starting with that letter. As more characters are added to the substring this variety decreases until a segment boundary is reached at which point this variety suddenly increases. Once the successor varieties of a word are

determined, they are used to segment the word and extract the stem. An example by Al-Shalabi et al (2005) is shown in figure 1.3:

Test Word: READABLE

Corpus: ABLE, APE, BEATABLE, FIXABLE, READ, READABLE, READING, READS, RED, ROPE, RIPE.

As can be seen, for the word 'Readable', starting with the first letter, the SV falls as each subsequent letter is added and then suddenly shoots up. This substring signifies it as a possible stem.

Prefix	SV	Letters
R	3	E,I,O
RE	2	A,D
REA	1	D
READ	3	A,I,S
READA	1	B
READAB	1	L
READABL	1	E
READABLE	1	BLANK

Figure 1.3: SV of a word

Co Occurrence Analysis: Co occurrence analysis is a means of measuring the occurrence of two words which occur close to each other in the text collection. If the occurrence of two words is in close proximity to each other, more than would be possible by chance, they are put into one cluster.

Xu and Croft (1998) proposed a promising language-independent technique to build or refine stem classes using co occurrence classes. They have stated that refinement can be done in the case of light or heavy stemming. Light or weak stemmers produce more stem classes failing to group related words. While strong or heavy stemming tends to create few class with words not related getting grouped together. No stemmer is perfect. Therefore in order to refine the classes, Xu and Croft employed a corpus analysis approach which is particularly effective in splitting up classes formed by strong stemmers. The classes get re-clustered based on a co occurrence measure. Their results showed that by applying the technique to a good light stemmer, further refined classes are obtained. Also, by applying to a crude strong stemmer stem classes were obtained which worked well. They tested their technique for English and Spanish which showed improvement in retrieval effectiveness.

String Similarity Measures using N-grams: N-Gram is a popular technique used extensively in Natural Language Processing (NLP), including text categorization, degraded text recognition, spoken document retrieval and text searching and retrieval (Mustafa, 2005). Here we would like to see its application in index term conflation and document retrieval.

The idea of using n-grams for retrieval was first put forward by Adamson and Boreham (1994). Their viewpoint was that the character structure of semantically related words is quite similar. Hence this technique would be useful for automatic classification of words. This technique has several advantages over light stemming, morphological analysis and other affix removal approaches. The latter suffer from the problem of being vulnerable to different regional spellings, misspellings, historical variant and random character errors, whereas these morphological variants are catered for by the string similarity measures (Mustafa, 2005).

In this approach, words are clustered together based their similarity to each other using N-gram matching. An N-gram is an N character window of a string. The similarity between a pair of words is calculated in terms of the number of common N-character strings between the two words. A certain ranking or threshold value is set, which helps to group the words containing most number of identical character substrings. Choosing the right value of N and whether to consider contiguous or non-contiguous character N-grams is what needs investigation. Different studies have been conducted with different choices made by researchers, some even taking into account the differential weighting of N-Grams to make the technique morphologically sensitive.

1.4 TREC Dataset for Evaluation[4]

The Text REtrieval Conferences (TREC) is co-sponsored by the National Institute of Standards and Technology (NIST) and U.S. Department of Defense with the aim to support research in the information retrieval community by providing the infrastructure necessary for large-scale evaluation of text retrieval methodologies. It is overseen by a program committee consisting of representatives from government, industry, and academia.

For each TREC track, NIST provides a dataset of documents and topics to run on the dataset. Participants of the conference run their own retrieval systems on the data, and the retrieved top-ranked documents are judged against relevance judgments for correctness, and evaluation of the results. The TREC test collections and evaluation software are available to the retrieval research community at large, so organizations can evaluate their own retrieval systems at any time. They are large enough so that they realistically model operational settings. Most of today's commercial search engines include technology first developed in TREC.

The evaluation effort has been growing over the years as more and more groups representing numerous countries are participating in growing number of tasks each year. TREC has also included the first large-scale evaluations of the retrieval of non-English (Spanish, Chinese and Arabic) documents, retrieval of recordings of speech, and retrieval across multiple languages. In the year 2003, ninety-three groups participated representing 23 countries for task carried out in various languages. TREC has thus successfully met its dual goals of improving the state-of-the-art in information retrieval and of facilitating technology transfer with retrieval system effectiveness approximately doubling in the first six years of TREC.

[4] From http://trec.nist.gov/overview.html.

1.5 Retrieval Evaluation

1.5.1 Recall-Precision Graphs

The most commonly used measures for evaluation of retrieval effectiveness are **recall** and **precision**. Recall is the ratio of the number of *relevant* documents *retrieved* to the number of total of *relevant* documents in the database. Precision is the ratio of the total number of *relevant* documents retrieved to the total number of documents retrieved. Recall basically tells us if the required relevant documents have been retrieved. Precision tell us a measure of how many of those documents retrieved are actually valid and not irrelevant. Let us denote the number of relevant documents *not* retrieved, the number of relevant documents retrieved, and the number of *irrelevant* documents retrieved, as X, Y and Z respectively, as seen in figure 1.4. The total number of relevant documents in the database would be X+Y, and the total number of documents retrieved would be Y+Z. The values of recall and precision would then be:

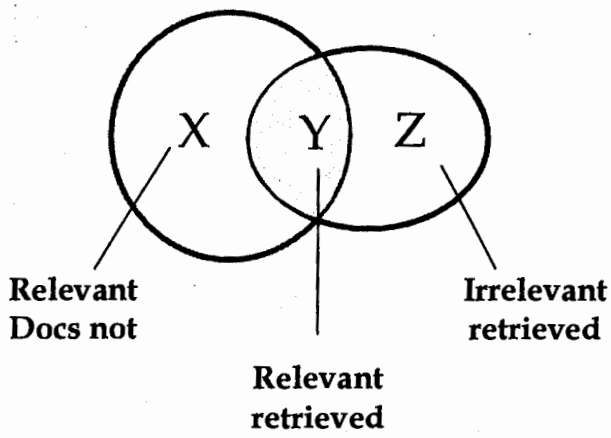


Figure 1.4: Measuring Retrieval Evaluation

$$\textbf{Recall} = \frac{Y}{X+Y}$$

$$\textbf{Precision} = \frac{Y}{Y+Z}$$

In order to view the results of precision and recall simultaneously, a bivariate graph is used with recall plotted against precision. This

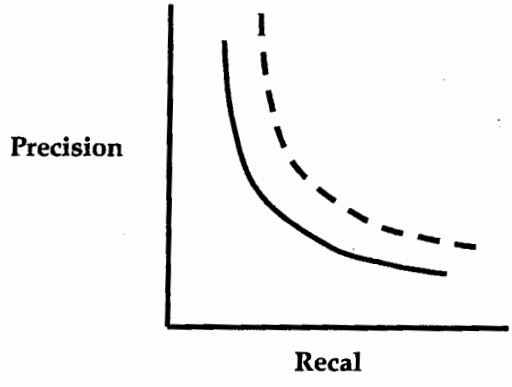


Figure 1.5: Plot of Recall vs Precision

plotting can be seen on previous page in figure 1.5. As can be seen, the two measures are inversely proportional. As the recall goes up, precision tends to come down and vise versa. Also, an improved performance is seen if the graph shifts outwards away from the origin. This is visible by the dashed line in the figure. This dashed line shows higher precision and recall values which is an improvement over the performance indicated by the solid line.

1.5.2 Statistical Significance Test

Significance tests are used to gauge whether the improvement we see of one technique over the other is significant or just a matter of chance. They help to add value to our Recall-Precision graph by giving the exact measure of the significance of improvement. There are numerous tests available, some of which are used to compare two methods and some used to compare more than two methods. Most of these tests assume the distribution of the values for measuring performance to be continuous and normally distributed and are referred to as *parametric* tests. Other *non parametric* tests do not make these underlying assumptions.

Typical tests used for IR evaluation are the Sign test and the Wilcoxon Signed-Rank test, which are non parametric test used to compare two methods. The sign test looks at the sign of the difference in the score of two methods, ignoring its magnitude. If one method performs better than the other, more frequently than would be expected on average then this method can be considered to be superior. The Wilcoxon test takes into consideration, in addition to the sign, the rank of the absolute value of the difference between the scores of the two methods.

We have chosen to experiment with the Wilcoxon test as it is more powerful being indicative of the relative magnitude in addition to the direction of difference where as the sign test only considers the direction (Siegel & Castellan, 1988).

The mean average precision score for each query is used as the measure of comparison. It helps to have as many queries as possible for the analysis but as empirical studies show that 25 is a minimum, which in our case is just sufficient. If the scores of the queries are denoted by X_i and Y_i where i = 1, 2, …, 25 for each query then the difference D_i in the scores would be equal to $X_i - Y_i$ .The Z-value for a particular analysis is computed by the following formula:

$$Z = \frac{\sum R_i}{\sqrt{\sum R_i^2}} \qquad \text{where, } R_i = \text{sign}(D_i) * \text{rank}(|D_i|).$$

From a table the p-value for the corresponding Z-value is obtained. If the p-value is less than 0.05, then the difference in the two methods is considered to be significant otherwise the difference is not significant.

2. Literature Review

2. Literature Review

Research in Arabic IR is scant as compared to English with few standard evaluation datasets and those too, only recently becoming available. Much emphasis has been laid on stemming for Arabic as compared to English. Stemming does not extensively improve retrieval for English as it does in the case of Arabic; yet standard stemmers like the Porter or Lovin's stemmer are available for English but no standard has yet emerged for Arabic; hence the quest for improved and better stemming techniques are under investigation.

In this section we will be firstly reviewing research on stemming detailing work under each category of stemming techniques. This is followed by a review of IR experiments at the Text Retrieval Conferences (TREC).

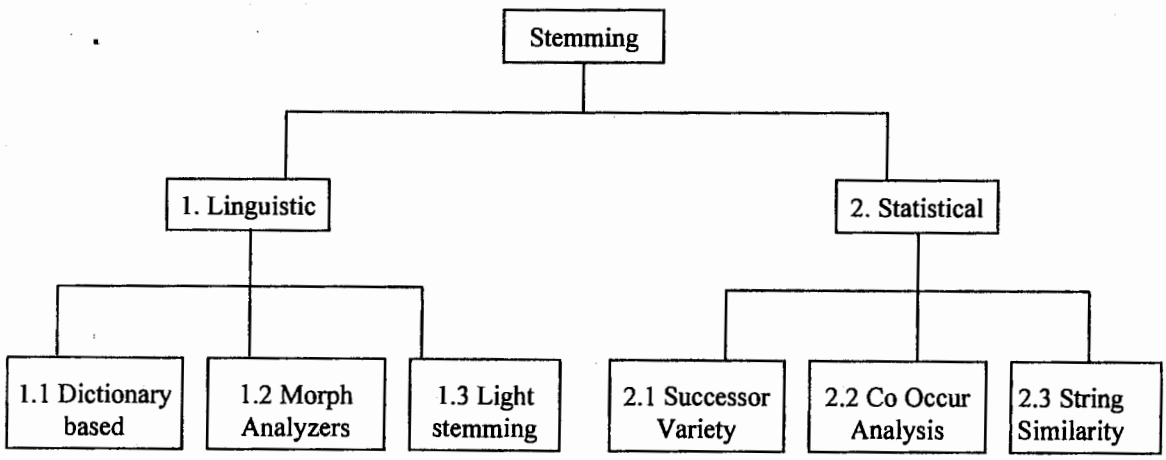


Figure 2.1: Classification of Stemming Techniques

Numerous researchers have worked on different stemming techniques which are categorized in figure 2.1. As stated earlier, we will be over viewing the main courses of research undertaken in the past especially highlighting those areas which form the basis of our research goal. In each category illustrated in figure 2.1, the most noteworthy work corresponding to each technique explored for stemming is discussed in this section.

2.1 Linguistic Approach

2.1.1 Dictionary Based

Al-Kharashi et. al. (1994), "Comparing words, stems, and roots as index terms in an Arabic information retrieval system"

In this early work on stemming the authors have shown the superiority of root/stem based indexing over words. They used a small text collection, constructed a larger *dictionary/ lookup table* storing Arabic words found in natural text with their corresponding Arabic parts (root, stem and affixation). Dictionary based approach involves building a large table storing Arabic words found in natural text with their corresponding Arabic parts (root, stem and affixation). Multiple entries may exist for a word with the same spelling to cope with possibility of multiple analyses i.e. different affixations to the same stem/root.

Although such dictionary/lookup table based approaches are accurate, it would be impractical to build the table for realistically sized corpora which are very large; there is too much linguistic effort to develop the table; besides, there are storage overheads and retrieval times needed for such data are long (Al-Sughaiyar, 2004).

2.1.2 Morphological Analyzers

Buckwalter, T. (2003), "QAMUS: Arabic lexicography"

An analyzer which returns stems was developed by Tim Buckwalter. The words are first segmented using some rules into three parts: prefix, stem and suffix. Dictionaries of prefix, stem and suffix are looked up to see if all the segments are present in their respective dictionaries. If this is so, next the compatibility of the segments with each other are checked using the compatibility tables. Finally the analysis is reported with each word tagged by its parts-of-speech (POS). In order to use the analyzer for retrieval

purposes, we simply take the stem from the initial segmentation phase and do not require the later stages of processing into POS tagging. An example of the output of the Buckwalter analyzer can be seen below:

Input Word: الشمالية → Al$mAlyp (Transliterated)
Analysis: (Al$amAliy~ap) [$amAliy~]
Al/DET + $amAliy~/ADJ + ap/NSUFF_FEM_SG

[$amAliy~] is the stem used for indexing. The rest of the POS tagging is not used for the purpose of IR.

Al-Sughaiyer, Imad A. et al. (2001), "Rule Parser for Arabic Stemmer"

In this article the authors discussed that the current morphological analysis techniques for Arabic language are based on heavy computational processes and the need for large amount of associated data. So utilizing these techniques greatly degrades the performance of information retrieval system. The authors proposed a new Arabic morphological analysis technique which based on the pattern similarity of words derived from different roots. This technique utilizes a very simple parser to scan coded rules and decompose a given Arabic word into its morphological components. For this new technique the authors showed good results without requiring complex computation.

2.1.3 Light Stemming

Darwish et al. (2002), "CLIR Experiments at Maryland for TREC-2002: Evidence combination for Arabic-English retrieval"

Al-stem was a light stemmer built by Kareem Darwish which removed some common prefixes and suffixes. It is available publicly for research purposes. In this paper it was compared to Leah Larkey's Modified U-Mass [2002] and was shown to be less effective

than it, but not significantly (see table in figure 2.2 below). Al-Stem used too many affixes for stemming. This resulted in heavier stemming.

	Mean Ave Precision	
	TREC 2001	**TREC 2002**
Al-Stem	0.286	0.316
Modified U-Mass	0.301	0.331

Figure 2.2: Comparison of U-Mass with Al-Stem

Mohammed Aljlayl et al. (2002), "On Arabic Search: Improving the Retrieval Effectiveness via a Light Stemming Approach"

Mohammed Aljlayl et al. (2002) presented two stemming algorithms for Arabic information retrieval systems and empirically investigated the effectiveness of both techniques. They showed that the root-based retrieval which based on the work of Khoja stemmer performs well than surface-based retrieval. But they pointed out that many word variants based on an identical root and creates invalid conflation classes that result in an ambiguous query which degrades the performance by adding irrelevant terms. So they proposed a novel light stemming algorithm for resolving the problem of ambiguity and showed that this light stemming algorithm significantly outperforms the root-based algorithm as shown in figure 2.3:

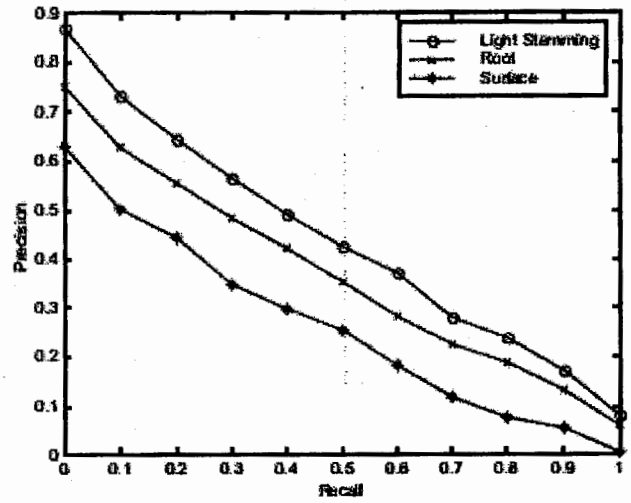


Figure 2.3: Comparison of Stemmers

Kazem et al. (2005), "Arabic Stemming Without A Root Dictionary"

In this paper the authors discussed the root-extraction stemmer developed by Khoja and pointed out some weaknesses in this stemmer. The authors implemented a root-extraction stemmer similar to Khoja's without requiring a root dictionary which can be difficult to maintain. Then they compared their stemmer to Khoja stemmer and Larkey's light stemmers (light8, light3) and found that it performs equivalently to the Khoja stemmer as well as "light" stemmers in monolingual document retrieval tasks performed on the documents collection as can be seen in figure 2.4. A root dictionary, therefore, does not improve Arabic monolingual document retrieval.

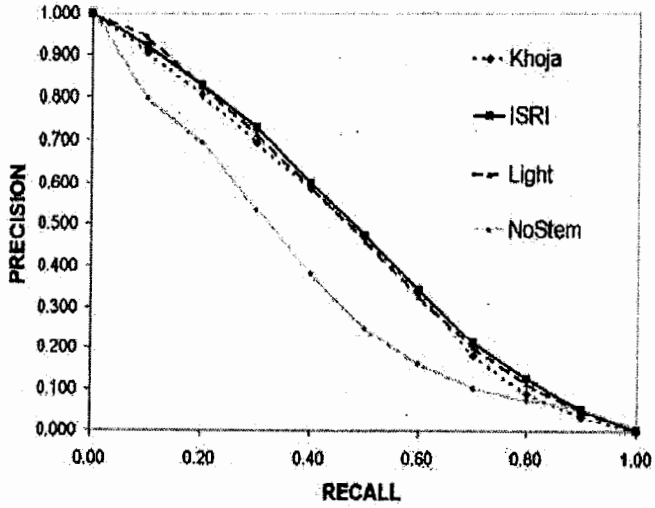


Figure 2.4: Comparison of Stemmers

Abdusalam F. A. Nwesri. et al. (2005), "Stemming Arabic Conjunctions and Prepositions"

Abdusalam et al. (2005) have proposed three novel approaches for dealing with conjunctions and prepositions in this article. Conjunctions and prepositions (Particles) are prefixes and Arabic has 9 conjunctions and 20 prepositions. The Approaches are:

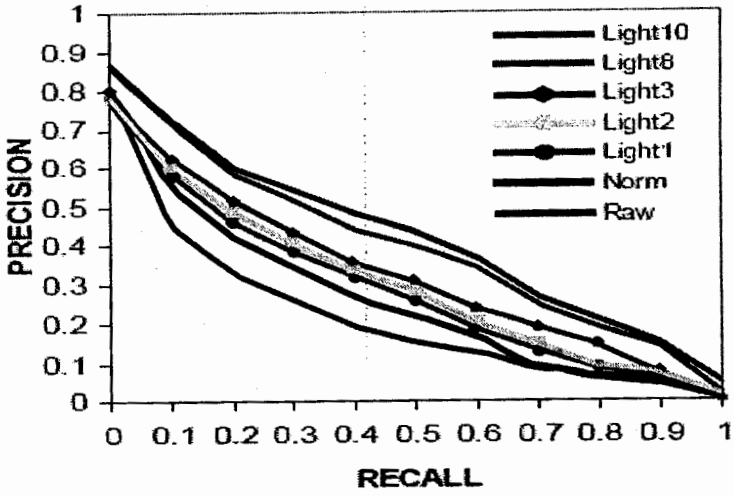
- If the first character matches a particle it will be removed if the remaining part is three characters or more--*Match and Truncate* (MT).
- If the first character matches a particle, remove it if the remaining part exists in the document collection--*Remove and Check* (RC).
- If the first character and the following characters form a well known prefix usually (*al*), remove it along with that prefix--*Remove With Other letters* (RW).

All the Existing stemmers use a combination of these approaches. These approaches focus on retaining valid Arabic core words, while maintaining high retrieval performance.

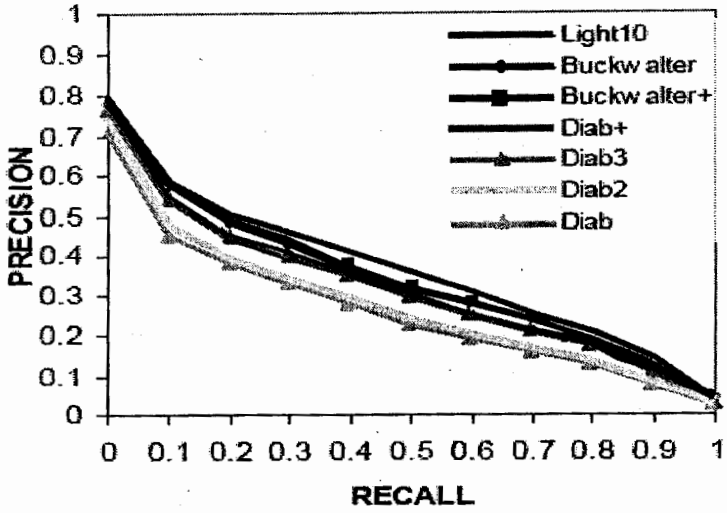
Larkey et. al. (2005), "Light Stemming for Arabic Information Retrieval"

Larkey et. al. (2005) have furthered their earlier work in which they compared four light stemmers (2002). In this paper (2005) they added another light stemmer light10. These five light stemmers were only different in the number of prefixes and affixes to be removed. These stemmers are compared in figure 2.5a. As can be seen that light10 gives the best result.

In this paper they also did comparisons with morphological analyzers. Their goal was to show the effectiveness of simple stemming without the need for considerable amount of linguistic knowledge and morphological manipulations. Using TREC-2001 data, their results showed that their best light stemmer, light10, which removed 7 prefixes, 10 suffixes and function words, gave better performance than the best analyzer, the Buckwalter analyzer, as seen in figure 2.5b.



2.5a: Comparison of Light Stemmers



2.5b: Comparison of Light10 with more Morph Analyzers

The comparison of light stemming to morphological analyzer shows that without trying to find roots or taking into account most of Arabic Morphology the light10 stemmer was at par with the best analyzer, Buckwalter, if not better.

Hayder K. Al Ameed et al. (2006), "Arabic Search Engines Improvement: A New Approach using Search Key Expansion Derived From Arabic Synonyms Structure"

Hyder K. Al Ameed et al. (2006) proposed a prototype which is an attempt to solve part of the complexities that Arabic language exhibits with the search engines and digital libraries. Arabic word affixation presents a dilemma for the users during the search process especially when it appears with different word patterns than search query pattern.

Another important point that is highlighted in the given paper is that, Arabic Information Retrieval (IR) systems practices are still bases on word-matching rather than word-sense approaches.

The study recommends the use of word stemming and wildcard search modules to solve the word scripts mismatching problem which arise with word-matching approach. It utilizes the synonyms facility in order to expand the queries in word-sense approach.

Momani M et al. (2007), "A Novel Algorithm to Extract Tri-Literal Arabic Roots"

In this article the authors implemented a novel stemming algorithm to extract tri-literal Arabic roots. They selected an Arabic text documents containing more than 1500 words and particles for testing the performance of the algorithm. They tested the algorithm in two runs, the first run contained diacritical text and the second run contained bare text. So these tests produced the proper roots with an accuracy of 73%.

2.2 Statistical Approach

2.2.1 Successor Variety

Al-Shalabi et. al., (2005) "Experiments with the Successor Variety Algorithm"

They have applied the successor variety technique to Arabic. In Arabic, unlike in English which has only suffixes, we must deal with both the prefixes and the suffixes. For the

suffixes the usual technique for English is applied. But for prefixes, the author has made a copy of the word corpus with the words in reverse order. The same successor variety technique can now be applied to the reserved word corpus yielding the segment boundary for the prefix.

The segment boundary was determined using the cut-off value method. Using this method a threshold for the number of successors is chosen. Those substrings with a value greater than the threshold are kept but later dropped from consideration if their occurrence in the text exceeds a value of 16 (i.e. they are just initial letters of the word). Hence we are left with two variables of a word, one with suffix removed and the other with prefix removed. Taking the intersection of the characters, we are left with the stem for indexing. They showed 80 % accuracy for this technique.

2.2.2 Co-occurrence Analysis

Larkey et. al. (2002), "Improving Stemming for Arabic Information Retrieval: Light Stemming and Co-occurrence Analysis"

As stated earlier, Xu and Croft (1998) first used this method to refine stem classes. Larkey et al applied Xu and Croft's co-occurrence method to Arabic. Their results showed that applying this technique to the stemmers did not improve retrieval effectiveness significantly as it did in the case of English and Spanish. The stem classes formed after applying refinement by co-occurrence analysis were an average size of 5 words per class, which is too small for Arabic. The technique worked for English, for which it was originally built, which has smaller stem class sizes due to fewer word variants but not for Arabic which tends to have larger classes due to abundance of variants for each word.

Their light10 stemmer showed much more improvement over this method. Light10 could not itself be improved further by this technique.

2.2.3 String Similarity

Mustafa et. al., (2004), "Using N-grams for Arabic text searching"

Mustafa and Radaideh used di-grams and tri-grams in order to find morphologically related words in the text. Their results showed superiority of di-grams over tri-grams. However, their tests showed that using N-grams based conflation to Arabic Corpus was inefficient. This they presumed is due to the lexical structure of the language which is highly inflected with extensive use of affixes.

2.3 Hybrid Approach

A hybrid approach combining the light stemming (linguistic), and the string similarity measure (statistical) was investigated by researchers, as seen in figure 2.6. These are discussed below in this section.

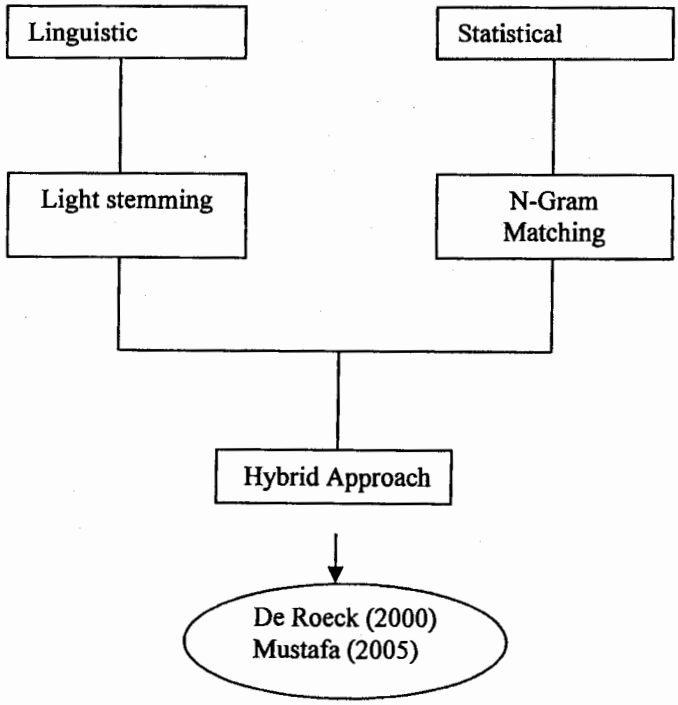


Figure 2.6: Researchers of a Hybrid Approach

De Roeck et. al., (2000), "A morphologically sensitive clustering algorithm for identifying Arabic roots."

A sophisticated hybrid approach using string similarity measures incorporating morphological sensitivity was developed by De Roeck and Al-Fares (2000). They made enhancements to Adamson and Boreham (1974).

Their approach is based on two stages. In the first stage light stemming was done. As they explained, affixes tend to decrease the similarity coefficient (SC) between words of same roots and increase the SC for words with different roots. In the second stage they performed a number of modifications to add morphological sensitivity to cluster the stems: (i) in order to minimize the effect of infixes, they have introduced the concept of a "cross" which is a non contiguous bi-gram of the character before and after a weak letter (alif, waw, ya). This is similar to the approach of Mustafa (2005), but different in that it does not include non contiguous bi-grams for all the character combinations. (ii) They have used differential weighting for bi-grams assigning a 0.25 weightage to bi-grams with weak letters and 0.50 weightage to bi-grams with potential affixes. The rest of the bi-grams have a weight of 1. (iii) Experimentally, they discovered that the best results are obtained using bi-grams, using single character overlap and blank insertions at word boundaries. Word boundary blanks helps to give full opportunity to boundary letters to contribute to SC. (iv) Finally, they have replaced Dice's formula to calculate SC with Jaccard's formula in order to cater for large impact of substrings given the shorter word length.

Using wordlist they showed up to 94% accuracy of their technique. They were unable to evaluate their performance in a search application due to non existence of standard datasets at the time.

Mustafa, S. H., (2005), "Character contiguity in N-gram-based word matching: the case for Arabic text searching"

Mustafa, S.H., besides experimenting with non-contiguous n-grams used *light stemming* before clustering. Using non-contiguous n-grams helps to eliminate the bi-grams involving weak letters which do not contribute to similarity value between words. They showed better performance using light stemming and non-contiguous N-Grams conflation.

2.4 Arabic IR at TREC

In the year 2001 for the first time a Cross Lingual Arabic language retrieval track was included in TREC with topics in Arabic, English as well as French. Since its inclusion more work on Arabic IR has taken place than had so far been possible in the past. Retrieval experiments were conducted using stems, roots and n-grams for indexing using machine translation, translation lexicons, parallel corpora and transliteration (Gey and Oard, 2002). The following is a brief list with short discussions of some of the IR research using the TREC 2001 Arabic track, (Abu El-Khair, 2003):

- Aljlayl et al. (2002) used the monolingual runs using roots and stems for indexing. These roots and stems were derived using a stemming algorithm developed by the author. They had better results with the stems than with the roots.
- Chen and Gey (2002) did experiments with monolingual and cross lingual runs. He showed better results for the cross-lingual runs over the monolingual runs.
- Darwish and Oard (2003) compared three stemmers in their monolingual runs, the light-8 stemmer, a slight modification of it, and Al-Stem which was developed by the author and modified by Larkey. There were small differences in the performance of the stemmers but they were not statistically significant.

- Larkey et al. (2002) performed experiments to compare different quality of light stemmers and co-occurrence analysis. They continued their work (2005) of comparison of their best stemmer, light 10, with different morphological analyzers. Their work has been discussed extensively in the section 2.1.3.

- Mayfield et al (2002) experimented with numerous character n-grams for indexing using a probabilistic retrieval system. Instead of words, various sized N-Grams were used for indexing. They investigated the use of 3-grams, 4-grams, and 5-gram and combinations of those n-gram lengths. The best results were obtained with a combination of 3-4-5 grams and words.

- Savoy and Rasolofo (2003) conducted a study comparing multiple weighting schemes from the Vector Space Model and the Okapi weighting scheme. The results indicate superiority of the Okapi function with light stemming. Further improvements of results were obtained with query expansion.

Although it has been increasing in the past few years, experimental research in Arabic Information retrieval is fairly limited compared to other languages. The previous review of experiments at TREC shows several stemming techniques that were utilized by different studies in the retrieval process are attempts to improve retrieval effectiveness. N-Grams for indexing have been successfully used Mayfield et al (2002) but so far no study has yet been carried out using N-Grams based word clusters for conflating Arabic roots as a stemming technique for Arabic using the TREC dataset.

2.5 Problem Identification

As seen in the literature, pure dictionary based/table lookup approach is impractical. A better use of lexicons was seen in the case of morphological analyzers which require less exhaustive linguistic effort in creating the word stems from different word variants.

Despite much pain staking work involved in building an analyzer we have seen that the simple light stemming approach can perform at par with the best and more sophisticated technique of morphological analysis. It offers a highly effective and simple approach to conflation.

However light stemmer only caters to the incidence of suffixes and prefixes by their removal. Infixes are not handled by lighter stemmers because they are indistinguishable from consonants. Also broken plurals do not obey normal grammatical rules and hence these also cannot be handled by light stemming. String similarity using n-grams approaches cater to these issues since many of the characters are the same in the variants of related words. (Xu et. al., 1998)

The work by Mustafa and Radaideh (2004) showed pure n-gram based matching does not give good performance for Arabic. Given the highly inflectional nature of the language, morphological variation cannot be ignored and need to be incorporated in some way.

We felt that the hybrid approach which combines light stemming and N-Gram matching as used by Mustafa (2005) and De Roeck (2000) is the most promising approach to cluster words variants and it is this approach that we chose to investigate further. So far there has been no document search evaluated on this technique due to non-availability of standard Evaluation Dataset in the past. In order to gauge the true standing of the Hybrid Stemming approach using document retrieval evaluation measures we have implemented an IR search engine using the evaluation resources of the standard Arabic TREC 2001 Dataset.

2.6 Objectives of the Research

Our objectives which we wish to achieve in this research are:

(1) To gauge the effectiveness of the Hybrid approach by *designing an Information Retrieval System (IRS) based on Hybrid Stemming approach* using the standard

evaluation TREC dataset which is sizable corpus of documents, as opposed to simple words retrieval being evaluated as previously used.

(2) In addition we wanted to evaluate the following *parameters for N-grams*:

- Formula for calculation of Similarity Coefficient (SC)
- Use of boundary space N-grams
- Character Contiguity of N-grams
- Differential weighting of N-grams
- Different values of the threshold

(3) We wanted to do a *comparison between simple stemming and the hybrid approach* in order to see the significance of improvement of the hybrid approach over simple technique of light stemming.

2.7 Justification for Our Work

The hybrid approach as used by Mustafa (2005) and De Roeck (2000) is the most promising approach to cluster words variants and this approach has so far not been implemented on IR application as made clear by the following statements:

- **Email from Anne De Roeck (Jan 2007):**
  "We were not able to evaluate a search or IR application...there are now datasets that can be used for testing retrieval performance."
- **Mustafa wrote in his paper:**
  "The focus of the present research was on string matching using a word oriented approach, the findings should not be taken further to conclude that text retrieval will necessarily exhibit the same pattern of behavior... this issue merits further investigation using a large document corpus."

3. Methodology

3. Methodology

3.1 Design of the Hybrid Arabic Information Retrieval System (HAIRS)

HAIRS, as we named our IR system, was built using the hybrid light stemming and N-Grams string similarity based stemming approach. It is based on the model of a typical IRS with additional N-Gram processing functionality with query expansion built into the retrieval process. Since our research focus was an evaluation of an existing technique we needed to choose a sizeable corpus which is a known standard for IR based research. We used the TREC Arabic 2001 track which is currently the only widely used standard dataset available for Arabic IR system evaluation.

For the basic functionality such as building the inverted index and doing ad hoc retrieval, we used an information retrieval toolkit known by the name of LEMUR which can handle Arabic document-based retrieval.

In this section we will first give an overview of the TREC Dataset outlining different metrics and available resources of evaluation from TREC. This is followed by a brief description of the LEMUR Toolkit and the functionality of the underlying API upon which our program modules were built. Finally we explain our system model built using the N-Grams approach and its implementation using functionality of the LEMUR API. The system model initially given in the proposal was slightly modified due to certain intricacies in the initial approach not visible at the time, without compromising on any objective of our research stated in the problem statement.

3.1.1 Arabic TREC 2001 Evaluation Dataset

The Arabic TREC 2001 dataset is the only standard available dataset that is used widely for the purpose of Information Retrieval. It is distributed by the Linguistic Data

Consortium (LDC) as Catalog Number LDC2001T55. Although available at a cost ($1200), LDC was kind to offer it to us under an evaluation-only license at no cost for a period expiring in June 2008.

The track was targeted at a wide audiences carrying out research using monolingual and cross-lingual IR as well as other NLP centric investigation. We used the resource for monolingual Arabic IR testing. The evaluation resources include Arabic documents, topic descriptions, relevance judgments, and Arabic Natural Language Processing tools. Details of the resource were outlined by Gey and Oard (2002):

Documents: The collection contains 383,872 Agence France Presse newswire stories (896 MB, uncompressed) in Arabic dating from 13th May1994 to 20th December 2000. This doc-base was created by David Graff and Kevin Walker at LDC. The source material was tagged using TPSTER style SGML and transcoded to Unicode (UTF-8). Each document is short with an average document length of 155 words. There are approximately 80 million words in the entire doc-base of which approximately 1 million are unique terms. (Sample document can be seen in Appendix A)

Topics: Twenty-five queries or topic descriptions were developed in Arabic by native Arabic speakers and then translated into English and French at NIST and were included in the TREC 2001 track. Further fifty queries were added to the topic set in TREC 2002 track. The topic descriptions include a very short "title" field designed to be representative of a query issued in an information retrieval system. A more extensive "description" field designed to be representative of what might be offered as an initial specification of what is needed to a search intermediary such as a librarian, and a "narrative" field that is intended as a detailed guide for assessing the relevance of individual documents.

We used the TREC 2001 track consisting of 25 queries using only the topics as queries to evaluate HAIRS. These are listed in Appendix B.

Relevance Judgments: In order to judge the validity of a search whether the documents retrieved are relevant to a given information need(topic) it is important to have human judgments of the relevance of documents to topics which provide the ground truth against which the effectiveness of an information retrieval system can be judged. These relevance judgments list the Boolean relevance of each query to all relevant documents in the doc-base to that query. By Boolean relevance we mean that it either lists whether a document is relevant or irrelevant without specifying any measure of the degree of relevance. (See sample in appendix C)

Pooling: Since it would not be practical to assess the relevance of every document to every query, so a purposive sampling method known as pooled relevance assessment is used in TREC. Participating systems are asked to rank the documents in order of decreasing likelihood of relevance to a topic, and a pool of documents to be judged is then created by combining the top-ranked documents from each system and removing duplicates. Documents that are not judged in this process are treated as if they are not relevant when computing retrieval effectiveness measures. Although this only approximates that value for a measure that would be obtained if complete relevance assessments were available, the approximate values do provide a useful basis for comparing systems.

TREC Evaluation Software[5]: Trec_eval is a program to evaluate TREC results using the standard, NIST evaluation procedures. The format for the command line is:

trec_eval [-q] [-a] trec_rel_file trec_top_file

Where 'trec_eval' is the executable name for the program, -q is a parameter specifying detail for all queries, -a is a parameter specifying summary output only (-a and –q are mutually exclusive), 'trec_rel_file' is the qrels, trec_top_file is the results file. The results file has the format: [Query_id, iter, docno, rank, sim, run_id] delimited by spaces. 'Query id' is the three digit query number (an integer) from 1-25 of the twenty-five queries, in our case. The 'iter' constant, 0, is required but ignored by 'trec_eval'. The

[5] Extracted from http://www.ir.iit.edu/~dagr/cs529/files/project_files/trec_eval_desc.htm

Document numbers are string values like '19940520_AFP_ARB.0017' (found between, <DOCNO> tags in the documents, sample in Appendix A). The Similarity, 'sim' is a float value. Rank is an integer from 0 to 1000, which is required but also ignored by the program. Run_id is a string which gets printed out with the output. An example of a line from the results file:

7 0 19940520_AFP_ARB.0017 1 42.38 *run-name*

Input is assumed to be sorted numerically by query_id. Sim is assumed to be higher for the docs to be retrieved first. Relevance for each 'docno' to 'qid' is determined from 'text_qrels_file', which consists of text tuples of the form [qid, iter, docno, rel] giving TREC document numbers (a string) and their relevance to a query. Tuples are assumed to be sorted numerically by 'query_id'. The text tuples with relevence judgements are converted to TR_VEC form and then submitted to the SMART evaluation routines to output the different evaluation measures.

Procedure is to read all the docs retrieved for a query, and all the relevant docs for that query, sort and rank the retrieved docs by 'sim'/ 'docno', and look up 'docno' in the relevant docs to determine relevance. Queries for which there are no relevant docs are ignored (the retrieved docs are NOT written out).

Explanation of the Official Values Printed:

The evaluation output by the program for an example run is shown in Appendix D. A brief explanation of each measure follows:

1. *Total number of documents over all queries*
   Retrieved: 25000
   Relevant:
   Rel_ret: (relevant and retrieved)

All values are totals over all queries being evaluated. Since there are 1000 documents retrieved for each 1000 there are total 25000 retrieved docs. The number of relevant is obtained from the relevance judgments. Finally, the Rel_ret are the relavant retrieved calculated from both the file.

2. *Interpolated Recall - Precision Averages:*

at 0.00
at 0.10
...
at 1.00

We have already given details of recall-precision evaluation in section 1.5.1. "Interpolated" means that, for example, precision at recall 0.10 (i.e., after 10% of rel docs for a query have been retrieved) is taken to be *maximum* of precision at all recall points >= 0.10. Values are averaged over all queries (for each of the 11 recall levels). These values are then used to form Recall-Precision graphs.

3. *Average precision (non-interpolated) over all relevant documents.* The precision is calculated after each relevant doc is retrieved. If a relevant doc is not retrieved, the precision is 0.0. All precision values are then averaged together to get a single number for the performance of a query. Conceptually this is the area underneath the recall-precision graph for the query. The values are then averaged over all queries.

4. *Precision:*

at 5 docs
at 10 docs
...
at 1000 docs

This measure gives the precision after X documents (whether relevant or non-relevant) have been retrieved. Values averaged over all queries. If X docs were not retrieved for a query, then all missing docs are assumed to be non-relevant.

5. *R-Precision (precision after R (= num_rel for a query) docs retrieved)*: This is a new measure, intended mainly to be used for routing environments and has been ignored from consideration in our case.

3.1.2 LEMUR-The Information Retrieval toolkit

Lemur is a toolkit designed to facilitate research in language modeling and information retrieval (IR). It was developed by collaboration between the Computer Science Department at the University of Massachusetts and the School of Computer Science at Carnegie Mellon University. The system's underlying architecture was built to support the technologies such as ad hoc and distributed retrieval, with structured queries, cross-language IR, summarization, filtering, and categorization.

Arabic language capability including the light 10 stemmer was recently added to the Toolkit by Leah Larkey. This addition has eased the development of an Arabic retrieval system for research and experimentation. The Toolkit uses Windows Code Page 1256 encoding (CP1256).

It supports the construction of basic text retrieval systems using the traditional methods such as those based on the vector space model and Okapi. The toolkit is available on the Web as open source software which makes it possible to freely modify it if necessary. The Toolkit code is written in C and C++, and runs on both UNIX and Windows (NT).

3.1.2.1 Lemur API[6]

The Lemur Application Program Interface (API) is intended to allow a programmer to use the toolkit for special-purpose applications that are not implemented in the toolkit itself. The API provides interfaces to Lemur classes that are grouped at three different levels:

Utility Level: This includes some common utilities such as memory management, a default exception handler, and a program argument handler. These utilities are useful even if a program is not doing language modeling or text retrieval. There are also several classes that support flexible document parsing. The main class is *TextHandler* which

[6] Extracted from http://www.lemurproject.org/lemur/api.php

allows for the chaining or pipelining of common parser components. The abstract class *DocStream* makes it easy to incorporate a user-defined document stream handler into the Lemur indexer, thus supporting different document formats. A basic document stream handler that recognizes a pre-defined document format is also included.

Indexer Level: An indexer converts a raw text collection to efficient data structures, so that the information (e.g., word counts) may be accessed conveniently and efficiently later. This indexer level depends on the utility level mentioned above. It provides a "push" method where the application is responsible for sending the appropriate information to an indexer through the *PushIndex* interface. The indexer level also contains an abstract Index interface for the access of indexed information, two implementations of this interface, and various kinds of indexing support, such as the common data structures used by Index: *TermInfo, TermInfoList, DocInfo,* and *DocInfoList*. This level supports not just retrieval but also any type of application that requires efficient access to the frequency information of words in a large text collection.

Retrieval Level: This level has abstract classes for general retrieval architecture and concrete classes for several specific information retrieval methods. Classes at this level are most useful for users who want to build a prototype system or an evaluation system as is the case with us.

3.1.2.2 Lemur's Basic Components and Working

Indexing and building Language Models: The figure 3.1 highlights some of the main components in the Lemur Toolkit and their interaction with each other. The arrows in the figure do not exactly indicate the input/output relationships; rather they are just intended to show how the components are built and depend on each other.

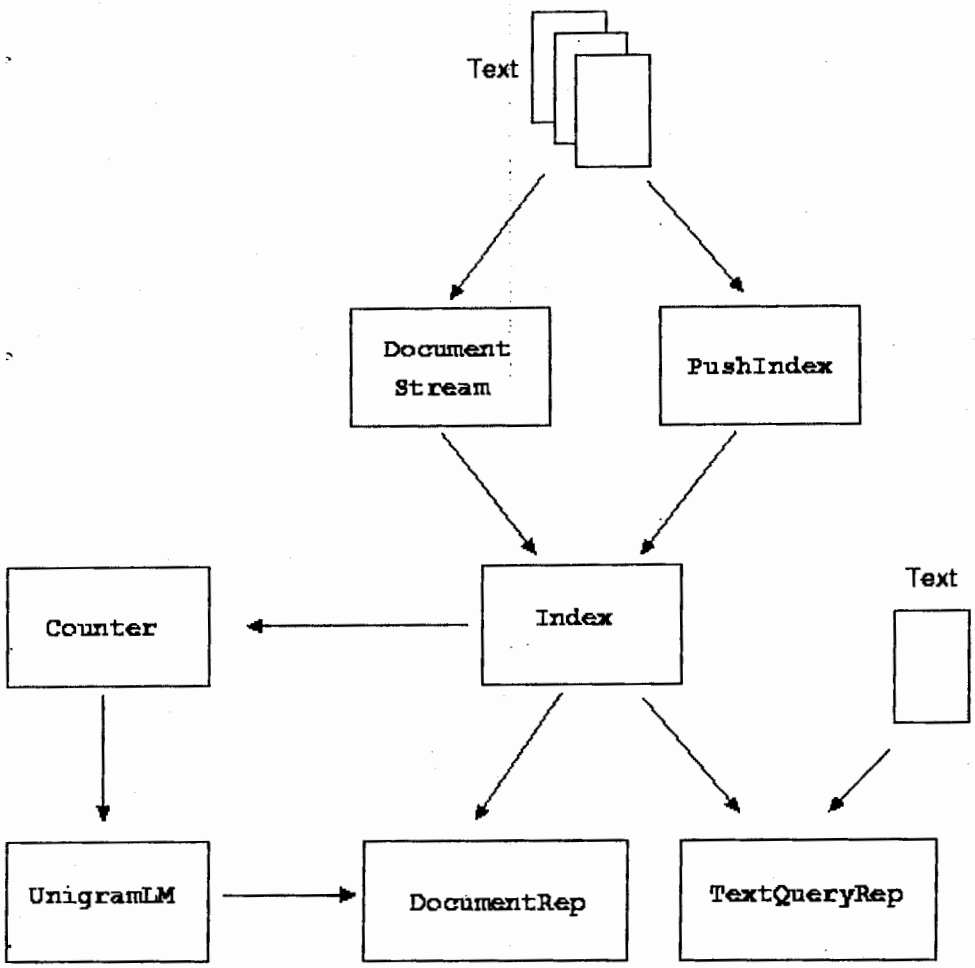


Figure 3.1: Class Dependencies for Indexing and Language Modeling

At the Indexer level, raw text of documents is transformed using the *DocumentStream* class into an abstract representation of a *document* used by the indexer. This transformation process includes tokenization and any required stemming or morphological processing that will be used by language models or retrieval methods. From this stream of documents the vocabulary of terms and their corresponding doc ids are extracted to build an inverted *Index*.

Another way of creating the index is through the *PushIndex* class. As the name infers, this class is as an API for 'pushing' documents and terms into an index rather than pulling them out of the document stream. The indexing application does the tokenization,

stemming stop-word removal of the tokens before sending them to the PushIndex to be pushed into the Index.

Once the index is created, Language Models (LMs) can be created from the documents in the collection. A *Counter* instance is created for a particular document which holds the empirical counts of the words (term frequencies) in the document. From the counts the actual model, *UniGramLM,* is created by applying an appropriate smoothing algorithm. This algorithm assigns an appropriate probability to each word. LMs can also be created indirectly by combining (smoothing) together other LMs.

Next we need a document representative and a query representative in order to perform the retrieval. These representatives are used to encode the particular way that a document or a query is represented in a particular retrieval method. Although built upon the same index or raw query each representation will be different according to the particular kind of retrieval method chosen. A *TextQueryRep*, which inherits from *QueryRep* is a representation of the weight terms of a text query. The *DocumentRep* and the *TextQueryRep* pre-compute the document collection statistics, such as inverse document frequency or backoff weights for smoothing. As shown in the figure the DocumentRep is built from the index and also depends on the UniGramLM for smoothing. The TextQueryRep is built from the raw text of the query but also requires the index in order to ensure consistency of the lexicon, tokenization, etc., and in order to have access to smoothing methods or inverse document frequencies.

Making a Retrieval Method for Text Queries: At the Retriever level, numerous retrieval methods are provided by the toolkit which all inherit from the class *TextQueryRetMethod.* This class implements a generic scoring function based on the *TextQueryRep DocumentRep* and a class *ScoreFunction.* The dependencies are shown in the figure 3.2.

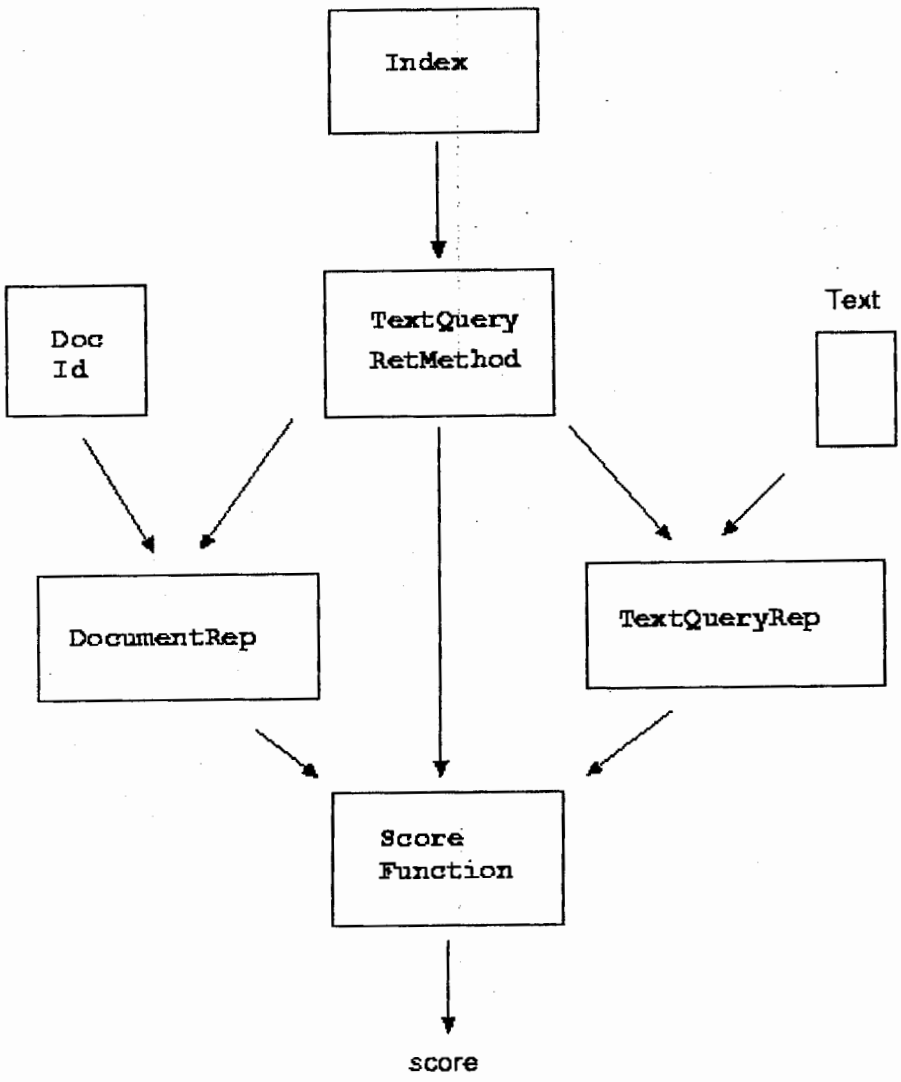


Figure 3.2: Class Dependencies for Document Retrieval

There are several functions that a *TextQueryRetMethod* must provide for efficient scoring based on the inverted index:

(1) Given a query it must provide to the *TextQueryRep*.

(2) Using the document id, it returns an appropriate *DocumentRep* instance to represent that document.

(3) Based the *DocumentRep* and the *TextQueryRep*, *TextQueryRetMethod* uses the *ScoreFunction* to compute the score of a particular document relative to that query. This score shows the similarity between the document and the query.

(4) Also provided by the *TextQueryRetMethod* is a query updating method used to update the query for relevance feedback.

Some of the *TextQueryRetMethods* that are provided by the toolkit are the basic TFIDF vector space model, Okapi, and a language modeling method using the Kullback-Leibler similarity measure between document and query language models.

3.1.2.3 Lemur Retrieval Methods[7]

TFIDFRetMethod is the vector space model that supports three different TF weighting methods for both query and document: raw TF, log-TF, and the BM25 TF. It uses the Euclidean dot-product as similarity measure. This model is not considered to be a well defined model as it contains certain heuristic values making it unsuitable to gauge retrieval performance efficiently. However the Lemur team implemented a variant of the TFIDF model based on the Okapi TF formula originally derived from a probabilistic model. A reasonable baseline performance can usually be obtained by using this variant with appropriate parameter settings. Details of implementation are available from Chengziang Zhai, 2001.

OkapiRetMethod is intended to be an exact implementation of the BM25 retrieval function as described in Robertson, 1994. The BM25 retrieval formula has a query TF for each term. In case the term is a new term extracted from the feedback documents, it does not have a natural "query TF". It is unclear from the original paper how this is set, so Lemur team has implemented it as an additional parameter.

SimpleKLRetMethod implements the KL-divergence retrieval model, which is an extension of the query-likelihood approach (Lafferty, 2001). It essentially scores a

[7] From section 2.8 on the webpage: http://www.lemurproject.org/lemur/api.php

document by computing the KL-divergence between the query language model and the document language model. Since they were only interested in ranking documents, the Lemur team rewrote the KL-divergence formula which only computes the part that affects ranking.

Which Model to Choose? This question has been researched in-depth by Abu Al-Khair, 2003 in this research thesis, "Effectiveness of Document Processing Techniques for Arabic IR" who explains his result as follows:

"To determine the most effective weighting scheme for Arabic language, three weighting schemes were used, TF*IDF weighting, Okapi best match algorithm, and the Kullback-Leibler Divergence Model as a language modeling approach. Without any additional linguistic processing the three schemes had a good performance with Arabic which was not surprising considering their previous success with other languages as they depend only on the corpus and query statistics. The differences between the three weighting schemes were very minimal and not statistically significant. The TF*IDF scheme is the best weighting scheme to be used with the Arabic language when used separately without stemming or stop words removal. This contradicts some previous research indicating that the BM25 algorithm is better when used with Arabic (Savoy & Rasolofo, 2003). One reason for this is that the term frequency portion of the TF*IDF scheme in Lemur is calculated using the term frequency portion in BM25 giving it the advantages of both of schemes. A second is that in Savoy and Rasolofo's experiment the BM25 was combined with a stemmer which particularly boosts the results with Arabic language."

We choose to use the probabilistic Okapi model which is the base of the TDIDF model and hence felt no need to use the TFIDF model. Given also that we would be using stemming before retrieval the Okapi model is more suitable for our work. The Okapi retrieval model has also been known to give better results by other researchers such as Savoy and Rasolofo, 2003.

3.1.3 The System Model

The Proposed (Initial) Model: Broadly outlining how we initially designed the IRS, the raw documents are placed in the data store. In the earlier model, three entities were proposed to be created and kept in the database: (i) an inverted file index, which stores the word stems and the corresponding document ids containing those stems (ii) Clusters, the stems with SC above a certain threshold (iii) Similarity Co efficient matrix, storing the SC value of all stems with each other. In order to create these entities there were three steps to be under taken, (1) Preprocessing, (2) Stemming and (3) N-Gram Matching. The model can be visualized in Figure 3.3.

1) Preprocessing: The text is first parsed to retrieve all the word tokens. The next step is normalization. A Number of steps are taken to normalize the words: (i) Remove diacritics (weak vowels), (ii) Remove non letters, (iii) Replace أ , آ , and إ with ا, (iv) Replace final letter ending in يي with ى, (v) Replace final ة with ه.

Finally, in this step is the stop word removal. These are the common words that appear in the text that carry little meaning. They can affect the retrieval effectiveness because they have a very high frequency and tend to diminish the impact of frequency differences among less common words, affecting the weighting process. The removal of the stop words also reduces the document length and subsequently affects the weighting process. They can improve efficiency due to the fact that they carry no meaning, which may result in a large amount of unproductive processing (Abu El-Khair, 2003). Examples of stop words are prepositions such as من , على , الى ,etc., which are ignored from the analysis.

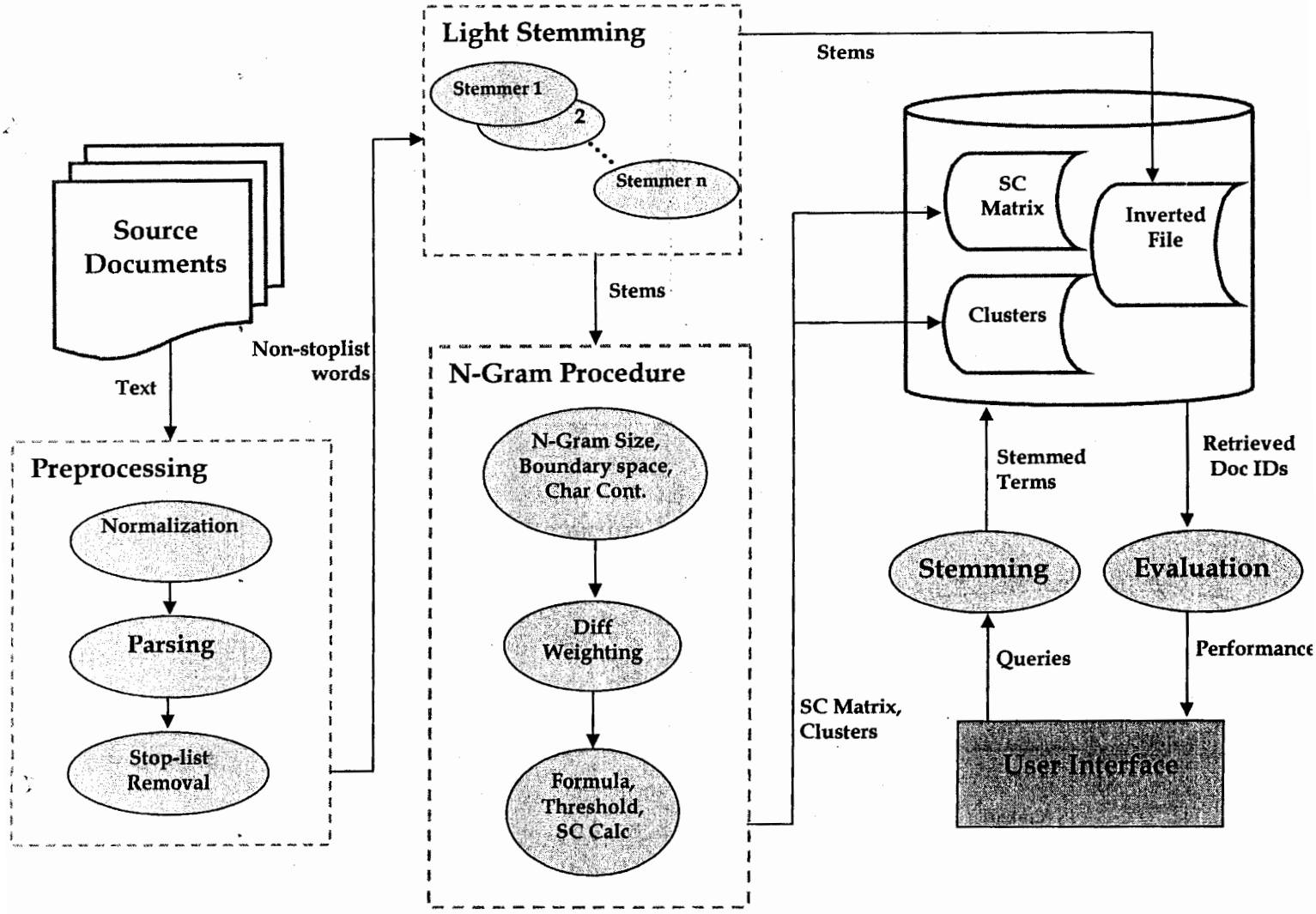


Figure 3.3: Initially Proposed Information Retrieval System Model

2) Stemming (Stage 1 of Clustering): This is the first stage in the clustering process. Different qualities of stemmers were to be applied to yield different stem class. The output of this stage is stems which are used to construct an inverted file index. Different indexes are built for the different quality of stemmers. Stems are also the input to the second stage of the clustering algorithm.

3) N-Gram Matching (Stage 2 of Clustering): This stage applies a number of steps as outlined in De Roeck's approach above. The outputs of this stage are the stem clusters and the SC Matrix which are stored in the database.

Searching: The user enters the query. The query terms are stemmed with the same stemmer that is applied in the clustering process. All the stems in the cluster of the stemmed query word are used in the document retrieval. Ranks are assigned to retrieved documents according to firstly, the number of exact matching stems and secondly, as the measure of the value of the SC amongst stems of the cluster. Higher rank will be given to documents containing stems with higher SC.

Limitations of the Proposed Model: We initially designed our system in accordance with the proposed scheme only to discover that there are major drawbacks in the model: The time and space requirements are too exorbitant.

The TREC dataset contains approximately 280,000 unique words. Each such word is matched with every other word to give its similarity score. Hence the total time needed to calculate the Similarity Matrix would be n^2. We noted that it takes approximate 2 minutes for a word to calculate its similarity with every other word on a P4, 1.3GHz machine, meaning thereby that it would take 2*280000 = 560000 minutes ≈ 389 days to calculate the similarity; more than one whole year! De Roeck (2000) applied this method to a small dataset of maximum 700 unique words for which it appears to be feasible. Although it is possible to build the matrix on distributed high speed machines but with limited resources at hand and given the fact that we are at this point evaluating the N-Gram approach requiring several SC Matrices to be compared it becomes impractical to adopt this course compelling alternative methods can be investigated.

Also, there are space limitations for producing such a giant Matrix. If the matrix is of float values each occupying 4 bytes the space requirement is again n^2. It therefore takes 280000*280000 = 78400000000 bytes ≈ 73 gigabytes of space for each matrix. A triangular matrix occupying half the space would reduce this to 73/2 = 36.5 GB. Since the

matrix is sparse, mostly containing zeros, a sparse matrix employing dimensionality reduction technique can drastically reduce its size to practical levels. However, this could perhaps slightly affect the efficiency of the system.

Although the actual retrieval process would not require extra time to give results, it is the initial building of the matrix that is time consuming. At this point the aim of this research is to discover the effectiveness of the N-Gram based retrieval technique employing several alternative strategies. Once the effectiveness of a particular technique is known to be significantly efficient and worth implementing it can be presented for high speed distributed processing to build a low dimensionality sparse matrix.

In order to achieve the objective of the research without the slightest compromise an alternative approach was taken to carry out the task of N-Gram based retrieval similar to the model proposed by Mustafa (2005).

The Revised Model: In this new scheme we decided to do away with the matrix calculation and instead built a weighted (similarity) graph of all the distinct terms in the topics (queries) with all the unique terms in the dataset. The system was designed such that if a new word is entered for searching in the query, first that new word is made part of the graph by calculating its similarity with all the unique words in the dataset which takes approximately 2 minutes per word to calculate. Each queried word once it became part of the graph would *not* be required to undergo the same expensive process of calculating similarities the next it appeared in the query. Thus each query word is first searched in the graph to see if it has already been processed before computing its similarity with all the words of the dataset. Although, in this scheme it would initially take quite long to do the actual retrieval, but as the number of queried words increases the subsequent queries, with repeating previously queried words, would take almost no time to form the expanded query for retrieval. The diagram in figure 3.4 shows the revised model for retrieval:

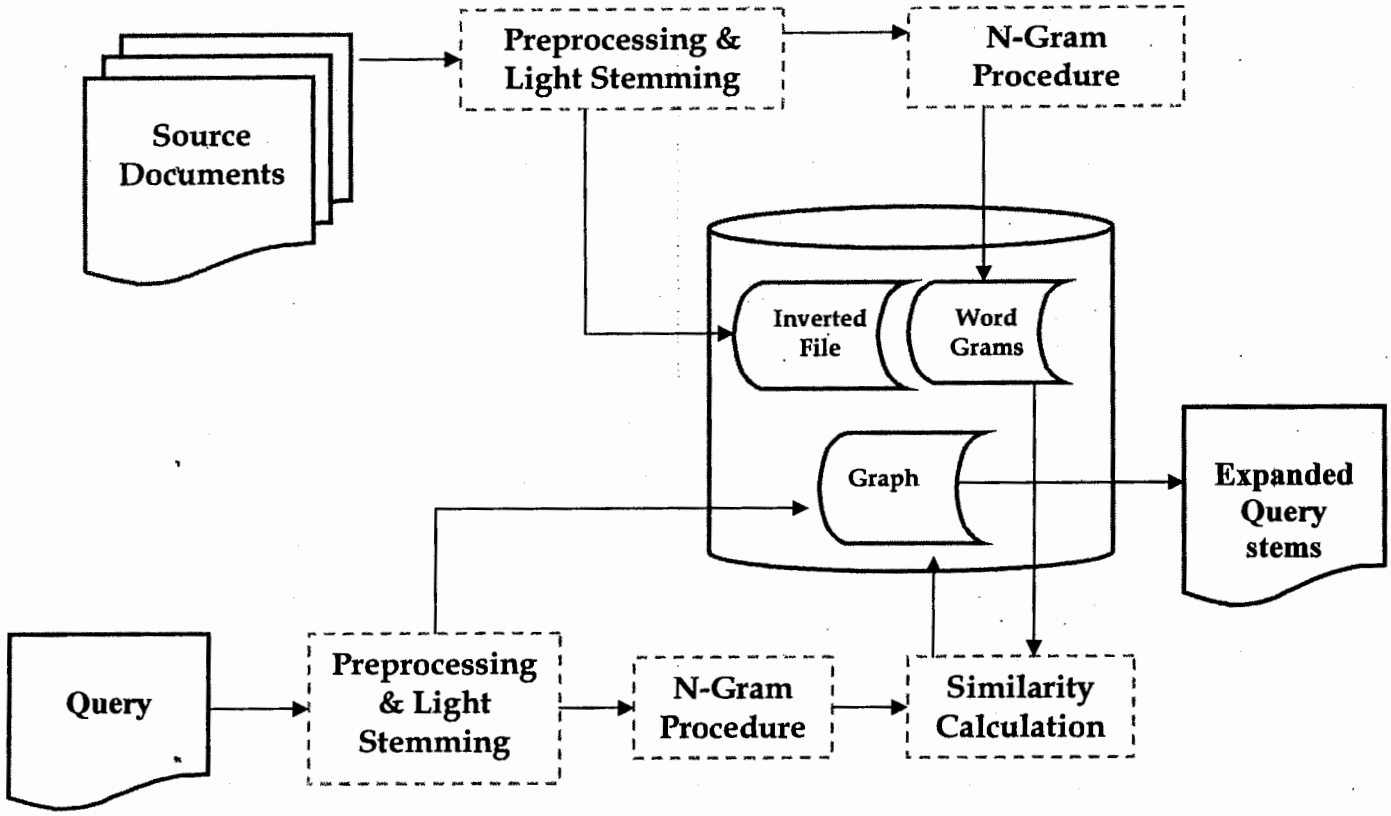


Figure 3.4: New Information Retrieval System Model

The pre-processing and light stemming stages are the same as previously explained in the initial model. The N-Gram procedure indicated in the diagram forms grams of the words depending on the type of technique employed, e.g. bi-gram, character contiguity, boundary space, etc. to create them. The grams are sorted and duplicates removed for matching the words. In calculating the similarity two measures were considered, the Dice's and Jaccard's formula. These give the value of the similarity based on the number of common unique n-grams in each word. Finally, once we have all the words of the query stored in the similarity graph, an expanded query is formed by selecting all the words related (syntactically similar) to each query term that are above a specified threshold. The expanded query is then used to retrieve the relevant documents using a particular retrieval method.

3.1.4 Implementation Details

In this section we explain the details of implementation of HAIRS using modules from the Lemur toolkit and additional functionality upon which the system was built.

Visual C++ 2005 was chosen as the platform for implementation as C++ is a powerful programming language. The VS 2005 is a tool supporting both managed and native code helping to provide integration with other related languages supported by the .NET framework. It also extends support for building sophisticated windows application using the managed components of the framework.

The Lemur API too was built for VS 2005. The size of the library is approximately 73 MB (in release mode). The toolkit comes with instructions to setup the library and start using the numerous applications that come built-in with the Toolkit. Amongst some of the applications that are provided by the toolkit are parsing, indexing, clustering, retrieval and retrieval evaluation.

3.1.4.1 Indexing[8]

Index: An index is basically a collection of information that can be quickly accessed, using some piece of information as a point of reference or key, which in our case are the terms in a collection of documents. These can be accessed later using either a term or a document as the reference.

Specifically, we can collect term frequency, term position, and document length statistics because those are most commonly needed for information retrieval. For example, from the index, you can find out how many times a certain term occurred in the collection of documents, or how many times it occurred in just one specific document. Retrieval algorithms that decide which documents to return for a given query use the collected information in the index in their scoring calculations.

[8] Taken and adapted from http://www.lemurproject.org/lemur/indexingfaq.php

Parsing and Document Format: Parsing refers to the process of breaking up text into tokens which are then processed to build terms. Before adding the term into the index, there might be some considerations to be made, such as whether or not that word is important enough to add, whether to add the word as is or to index its stem form instead, and whether to recognize certain words as acronyms. Having an acronyms list, ignoring stopwords (very common words, like "the", "and", "it"), and indexing word stems (so "stem", "stemming", and "stems" would all become the same term) are features supported by Lemur. These features are all supported by the provided application, BuildIndex.

Since Lemur is primarily a research system so the included parsers were designed to facilitate indexing many documents that are in the same file. In order for the index to know where the document boundaries are within files, each document must have begin document and end document tags. These tags are similar to HTML or XML tags and are actually the format for NIST's Text REtrieval Conference (TREC) documents.

The two most frequently used parsers are the *TrecParser* and *WebParser* provided by the toolkit. Since we are using TREC documents we used the *TRECParser*. This parser recognizes text in the TEXT, HL, HEAD, HEADLINE, TTL, and LP fields (see Appendix A for sample). For example:

```
<DOC>
        <DOCNO>
                Document number
        </DOCNO>
        <TEXT>
                Index this document text.
        </TEXT>
</DOC>
```

Type of Index: Lemur currently has two available index types: KeyfileIncIndex, and IndriIndex. The indexes are different in that they might index different data or represent the data differently on disk. Each index has a "table of contents" file which has some summary statistics on what's in the index as well as which files are needed to load the index. When we want to use an index, we will need its table of contents file to load it.

We have used the KeyFileIncIndex as it is the easiest to use and has other advantages: The index stores the indexed terms in an inverted list as well as the positions of the terms within the documents. This index can also have a document manager associated with it to store properties (metadata) about each document. The KeyfileIncIndex also has the additional ability to support incremental building by allowing new documents to be added to the index. A Keyfile index has the extention ".key" for its table of contents file.

Pre-processing and Light Stemming The Lemur toolkit provides support for Arabic parsing of text. The three basic options for parsing are: (1) Normalization (2) Stopword Removal (3) Light Stemming.

As discussed earlier the preprocessing phase constitutes the normalization and stopword removal. During normalization the parser removes diacritics, punctuations, non-letters and does certain substitutions, as discussed above, such as replacing the ٱ and ٲ with ا. The stopword list consists of 168 stopwords (given in Appendix F). As stated earlier, these are the common words that appear in the text that carry little meaning and hence can be ignored from the analysis. As for the light stemming, Lemur provides only one stemmer, the light10 created by Leah Larkey. This light stemmer has been gradually improved from its earlier version, hence the number '10' in the name. It has thus far been the best performer at TREC in comparison to some others like Darwish.

There are several combinations of options available to parse the text by specifying the appropriate parameter for the kind of processing desired. Referred to as the Arabic Stem Functions one of 5 options can be specified.

- arabic_stop : Arabic stop words removal
- arabic_norm2 : Table normalization
- arabic_norm2_stop : Table normalization with stopword removal
- arabic_light10 : Larkey's light10 stemming
- arabic_light10_stop : light10, normalization and removal of stopwords

The Lemur Indexing GUI

The index and inverted file was created using the LemurIndex GUI application, shown in figure 3.5, which uses the underlying *BuildIndex* application provided by the Lemur Toolkit.

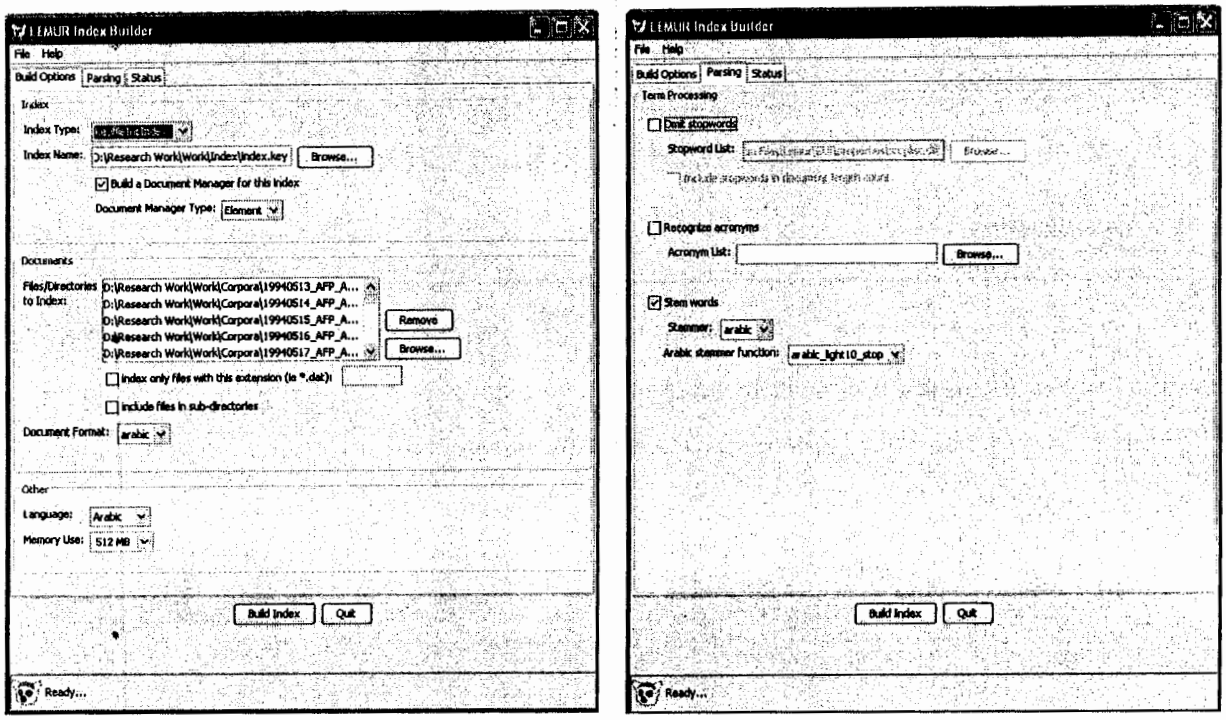


Figure 3.5: Indexing GUI

As visible in the figure, we created a Key File index and a Document Manager to handle the index. The document format was set to Arabic and the arabic_light10_stop stem function was chosen to stem the words.

Initially there were problems with the Indexing application. The indexing would be created perfectly fine without the using any stem function; but with the stemmer the application would fail. We wrote to the Lemur team who noted the bug in the program and rectified three key problems in the Arabic stemming code of Lemur. The new release Lemur 4.5 was the bug free version which we have used for indexing.

3.1.4.2 HAIRS Application

The HAIRS application consists of various classes of which the main classes are those responsible for creating word N-Gram tokens and calculating the similarity between words. In this section we will give the details of how the API accesses the index and details of the structure of the HAIRS application with explanations of the functionality of each module.

The application was built with the Lemur API, incorporating the lemur.lib file and other *include* files into the project. Since HAIRS application was in debug mode, the lemur.lib file also needed to be in debug mode; hence we created a new lemur.lib file from the Lemur source code, which was approximately 112 MB in size, larger than its size in release mode (72 MB).

Access to Index Information[9]

Once we have created the index, the Lemur API provides ease of access to the index. Functionality is provided giving access to terms of the index along with numerous statistics, such as accessing the unique terms, their unique term count, word frequency of occurrence etc. Some of the functions supported by Index can be grouped in the following way:

Open: An indexer must be opened before any access function can work. The open function takes a single argument: the name of the table of contents (toc) file that is created by an indexer.

Summary counts: There are various kinds of summary counts made available by the index-manager. The count of terms belongs to two types: regular counts and unique counts. The function name indicates this difference (i.e., termCount and termCountUnique). The regular counts include every occurrence of the token, while the unique counts ignore repeated occurrences of the same token. The counts supported include:

[9] Extracted from http://www.lemurproject.org/lemur/api.php

- Total number of documents (method *docCount()*)
- Total number of different/unique terms in the collection, i.e., the vocabulary size (method *termCountUnique()*)
- Total number of terms in a document (i.e., document length) (method *docLength(int docID)*)
- Total number of documents with a particular term (i.e., document frequency , e.g., used in the IDF formula) (method *docCount(int termID)*)
- Total count of occurrences of a term in the collection (method *termCount(int termID)*)
- Total count of all terms in the whole collection (method *termCount()*)
- Average number of terms in a document (i.e., average document length) (method *docLengthAvg()*)

Index access: The basic information that an indexer stores is a document by term matrix with each entry denoting the frequency count of a term in a document. There are two ways to access this matrix: by document or by term. That is, with document ID as the key, you can obtain a list of counts (called a *TermInfoList* in Lemur), each corresponding to a term occurring in the document. Alternatively, you can use a term ID as the key to obtain a list of counts (called a *DocInfoList* in Lemur), each corresponding to the count of the term in each document in which it occurs. An index that supports the second way is often referred to as an inverted index. We used the inverted index to access all the terms in the doc-base to perform N-Gram procedure on. Other functions relating to terms were also utilized as needed.

Grams and Word-Grams

Each word in the text is broken up into n-gram tokens in order to calculate its similarity with other words. A class *Gram* is used to represent a single token which consists of an n-sized character array and an associated weigh of the gram token. The default value of the weight of each *Gram* instance is 1; this weight is changed when forming differentially weighed N-Grams.

The class *WordGrams* holds a vector of Gram. This vector represents a textual word broken down into overlapping n-sized character grams. The *GetWieght()* function in this class returns the sum of the weights of all grams in a word.

N-Gram Procedure

The class, *NGramProcedure*, is responsible for the creation of the N-Grams. This class has been designed to be a generic abstract class for forming any kind of N-Grams of words. It basically consists of a list of gram tokens of a words, similar to the *WordGrams* class, which are processed by subsequent inherited classes to obtain a certain type of n-gram tokens depending on the procedure chosen. A single virtual function,

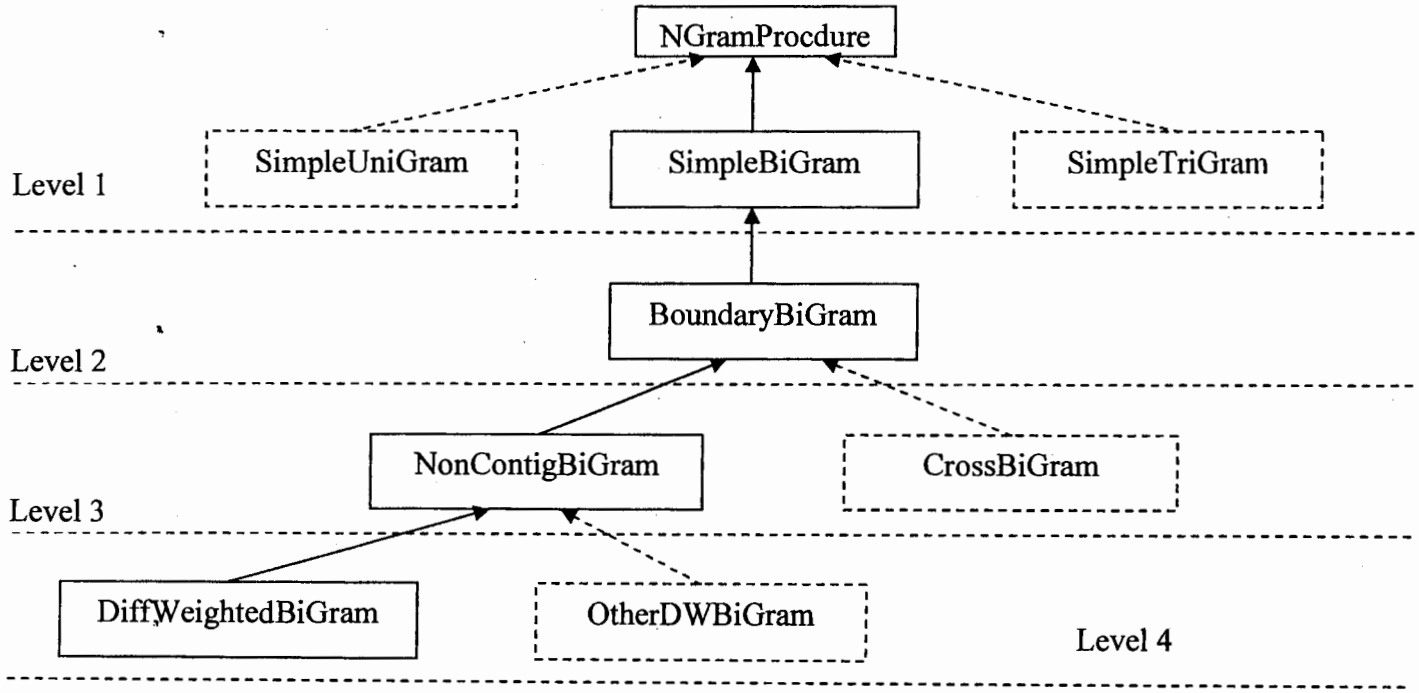


Figure 3.6: N-Gram Procedure Inheritance Class

get_grams(String) takes a word and returns its n-gram tokens. Subclasses inherit from this parent class depending on the type of N-Gram Procedure to be applied. The inheritance structure is shown in the figure 3.6.

The levels show at each stage that the list of grams is modified according to the procedure at that level. Thus at level 1 simple n-grams are created with a one character overlap window by the *simple()* function. These n-grams are modified to include a boundary space gram token by the *boundary()* function at level 2. The list of n-gram tokens is enhanced to include non-contiguous n-grams by the *noncontig()* function at level 3. Finally, at level 4 different differential weighting schemes are applied to the n-gram tokens by the *diffweight()* function.

Depending on the procedure being tested, n-grams are returned by the *get_grams()* function implemented by the class at each level. This function calls all the protected functions in the hierarchy above, involved in the making of n-grams. Thus, for example, *NonContigBiGram* class will call the *simple()* function of the *SimpleBiGram* class and the *boundary()* function of the *BoundaryBiGram* class in sequence before it calls its own *non_contig()* function, which itself is a protected member function called by subsequent classes at level 4.

The solid lines and boxes in the figure shows the part actually implemented; the dashed lines and boxes in the hierarchy structure show possible future directions of implementation and testing.

Similarity Calculation

SCMethod is an abstract class used to calculate the Similarity Coefficient (SC) of two words. The virtual function, *calc_sim(WordGrams, WordGrams)*, is used to calculate and

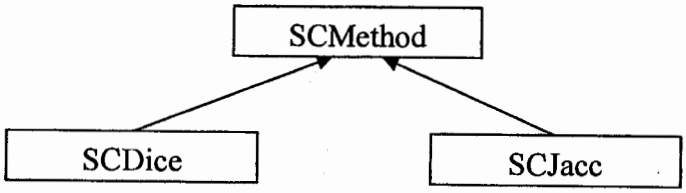


Figure 3.7: SCMethod Class Diagram

return the similarity score given the input of two words in the form of Word-Grams created by the *NGramProcedure* class. Two types of measures for calculating similarity were implemented, the Dice's measure and the Jaccard's measure shown in figure 3.7.

Both inherited classes calculate the similarity based on the count of unique common gram token in each word; the only difference is in the formula for calculating the SC value. Default threshold values are set to 0.75 for Dice and 0.5 for Jaccard. The higher value for Dice is because Dice's formula tends to raise the value of the SC whereas in the case of Jaccard lower SC values are obtained for the same two words. More measure of calculating similarity can be implemented as subclasses to the *SCMethod* class, such as the Manhattan distance measure etc.

Weighted Graph Data Structure and Related Classes

There are two ways of storing a graph. If the graph has edges from most vertices to every other vertex i.e. it is highly connected then a more efficient data structure for storing such a graph is an adjacency matrix. But if the graph is sparsely connected with few connections from a vertex to other vertices then it would be more efficient to store the graph in an adjacency list. In our case the latter is valid since a queried word will form similarities with only a very small subset of the total dataset vocabulary. We implemented the adjacency list in the form a B-Tree for fast lookup of words in the tree as compared to a list. The structure is shown in the figure 3.8.

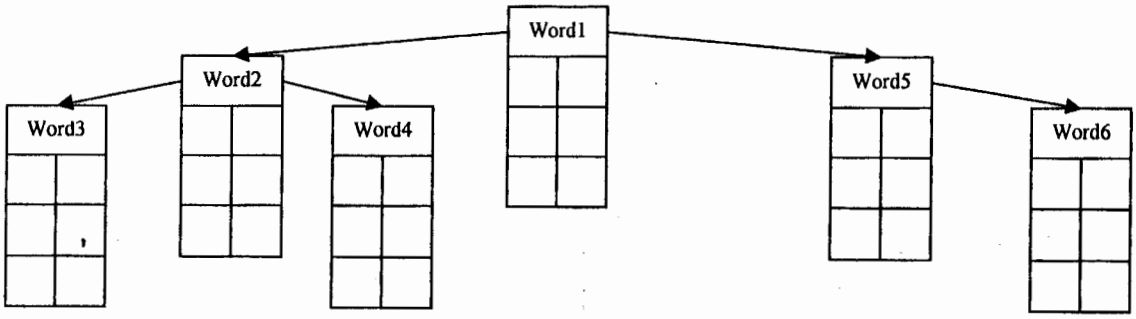


Figure 3.8: Binary Tree of Word Clusters

Whenever a word that has not been queried before is entered, its similarity scores with all other words are calculated and kept as an instance of class *WordCluster*. Each instance of

WordCluster has a queried word string part and a vector of all words which are similar to the queried word. Each similar word, represented as an instance of the class *SimWord*, is itself composed of an integer id of the word in the unique word vocabulary of the dataset and a score of similarity. A node in the tree, in figure 3.9, represents a *WordCluster* instance with each instance holding a word and a vector of *SimWord* instances.

WordClusTree class is used to maintain a B-Tree of type *WordCluster* and handle any lookup or insertion into the tree. A separate tree is used for each N-Gram Procedure type and similarity measure chosen. When the application starts the B-Trees are loaded from the file into memory. Whenever a query word is entered, it is first searched in the appropriate B-Tree to see if it exists. If the entry is found, a *WordCluster* instance is returned with the *SimWord* vector sorted on the similarity score. When the application quits, the B-Trees are stored using in-order traversal onto a file. The advantage of storing then in-order is that everytime the trees are being read from the file they are transformed into balanced B-Trees in memory.

Building a Word Cluster

The *WordClusBuilder* class is the engine for building the Word Cluster of a queried word. It has a handle open to the index in order to access the terms in the dataset. There are two basic functions provided by this class. First, is the *init_words (NGramProcdure, SCMethod)* function which, given the n-gram procedure to apply, creates the *WordGrams*

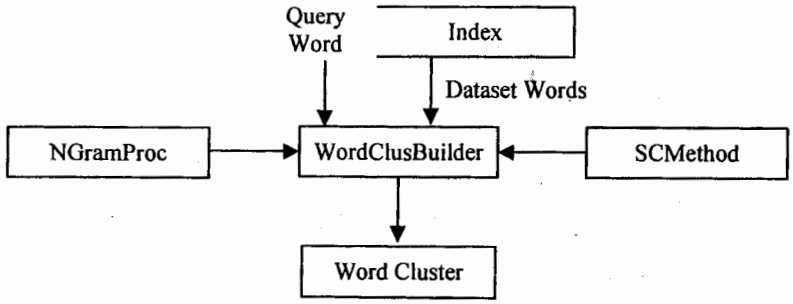


Figure 3.9: Class Dependencies for Building Word Clusters

instance of all the words in the dataset. The word grams are sorted and duplicates are removed to that time is saved later when the similarity score is being calculated. These

WordGrams instances are kept in a vector in memory. Secondly, the *get_clus(string)* function takes the query word as input, computes the SC score according to the chosen *SCMethod* subclass instance with all the *WordGrams* kept in the vector. Finally, the *WordCluster* formed from similar words that exceed the threshold value and returned. Note that it is not the responsibility of *WordClusBuilder* class to add this cluster to the weighted graph/tree; this is done elsewhere.

Query Expansion

The raw query is transformed into a stemmed expanded query using the *ExpQryBuilder* class. The appropriate B-Tree to be used is passed using the *WordClusTree* instance. The function, *build_expQry(Document)*, takes input of type *Document* (see Section 3.1.2.2). This query *Document* instance is then converted to a *TextQuery* object. *TextQuery* class inherits from *TermQuery* class in Lemur which is an abstract interface for a query containing a sequence of terms. Functionality for iterating over each term is provided by *TermQuery* class. Hence, each term is extracted and stemmed if it is not already stemmed. For each term of the query the tree is searched to see if the term has been queried before and hence would be present in the tree. If the term is found the *WordCluster* instance is returned with sorted the *SimWord* vector. An arbitrary value of threshold is set for the desired level of similarity. All words from the *WordCluster* are appended to the new query, which are above the desired similarity threshold. Numerous options can be chosen for expanding the query as outlined in section 3.3. If a stem is not

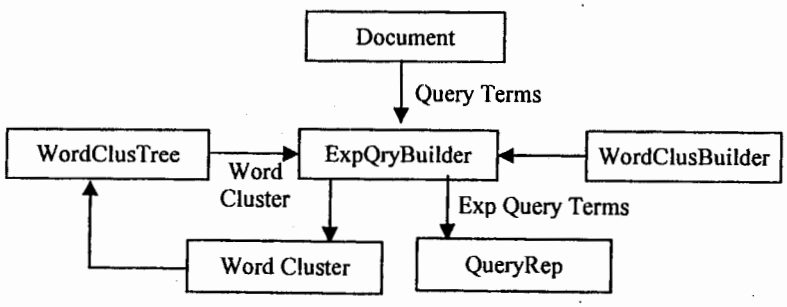


Figure 3.10: Class Dependencies for Building Expanded Query

found in the tree, then using an instance of the *WordClustBuilder* a *WordCluster* of the

query word is obtained which is used to expand the query, as well as inserted into the B-Tree using the instance of the *WordClusTree*. Once the string of expanded query terms is obtained, a *TextQueryRep* (see section 3.1.2.2) of the query terms is returned by the function *build_expQry()*.

Retrieval and Evaluation:

Retrieval of documents is performed by the *DocRetriever* class. This class has a function *RetEval()* which reads the topics file containing queries and outputs a result file used for TREC evaluation. A *TreePopulator* class is used to fill up the B-Trees with all the terms in the stemmed topics file for a particular N-Gram Procedure. This helps to concentrate on the retrieval computations saving time from computing SC values at the time of querying. As stated in section 3.1.2.3, we have used the Okapi retrieval method.

A *ScoreAccumalator,* which holds the score of each document that is score incrementally with respect to its query and document representative, is passed into the constructor of *okapiRetMethod* along with the handle for the index. The parameters used for Okapi retrieval method are the default with K1 = 1.2, B = 0.75 and K3 = 7. The description of these chosen parameters can be found in (Robertson et al, 1994).

Each retrieval method provides a function, *computeQueryRep*, in order to calculate the *TextQueryRep* of a given query. This computes a query representative according to the type of retrieval method chosen based of the term frequencies in the text query. We bypassed this method of computing the Query representative and instead created the representative manually, setting the weight of each term if required by using the *setScore()* function provided in the *ArrayQueryRep*, a subclass of *TextQueryRep*.

Once we have the query representative, we use the *ScoreCollection* function to score the entire collection. This function calls the appropriate *ScoreFunction* of the *okapiQueryRep* in order to score a document with respect to a certain query. The scored documents are kept in an *IndexedRealVector* object which is basically a vector of all document ids in the index and the corresponding score of each document.

Finally, once we have the documents with their corresponding score, the results are output into a result file for the top 1000 document scores for each query. The format of each line of the output file is

Q_Id 0 Doc_Num 0 Doc_score XXX

This document format conforms to the requirements of the TREC evaluation software. Although five fields are taken as input by the software only three are relevant and actually used to calculate evaluation parameters, the query id ('Q_Id'), the document id ('Doc_Num') and the document score ('Doc_score'). Details of using the TREC evaluation software are given in section 3.1.1.

3.2 Parameters for N-Grams

N-Gram matching for word conflation was first introduced by Adamson and Boreham (1994) who observed that words that are syntactically similar are also related semantically. They developed a technique to calculate similarity of two words as a factor of the number of shared sub-strings. Further refinements and modifications to the N-Gram parameter can shift its inclusion from being a purely statistical technique to one which exhibits morphological sensitivity. This approach is a promising approach for Arabic Root Based string matching and has been used by researchers giving positive results on word-list with varying parameter for the formation of substring. In this section we will explain these variations of N-Gram parameters which we have used in our work, particularly with reference to Arabic words.

The Original Simple Approach: In their original study, Adamson and Boreham developed an algorithm which drags an N-Sized window across two strings with a 1 character overlap, and removes duplicate N-Gram tokens. The Similarity Coefficient (SC) of the two strings was calculated using the Dice's equation: SC (Dice) = 2 * (number of shared unique n-grams) / (sum of unique n-grams in each string). An illustration of the algorithm is shown in the table 3.1.

Language	String	Bi-Grams	Unique Bi-Grams	Shared Bi-Gram	SC (Dice)
English	phosphorus	ph ho os sp ph ho or ru us	ph ho os sp or ru us (7)	ph ho os sp (4)	2*(4)/(7+7) = 0.57
	Phosphate	ph ho os sp ha at te	Ph ho os sp ha at te (7)		
Arabic	اسلام (Islam)	اس سل لا ام	(4) اس سل لا ام	سل لا ام (3)	2*(3)/(4+3) = 0.86
	سلام (Peace)	سل لا ام	سل لا ام (3)		

Table 3.1: Original Adamson's N-Gram String Matching

As can be seen from the table, words that are related syntactically are also related semantically giving a higher value of SC. A word's similarity to all other words in the dataset is calculated. A threshold level is chosen as a cut-off to choose only those words that are more similar to be grouped together. The higher the SC cut-off the fewer the number of words will get clustered together. If a low SC cut-off is chosen more unrelated words will get grouped together.

Simple N-Gram word matching has known to give positive results for English as shown by Kosinov (2001). He compared three stemming approaches for English: (i) Affix removal using the Porter stemmer, (ii) Successor Variety, with first, last and maximum peak cut-offs (iii) N-Gram matching with 2,3 and 4 –grams sizes. The figure 3.11 indicates that Bi-Gram word matching clearly outperforms affix removal and successor variety algorithm.

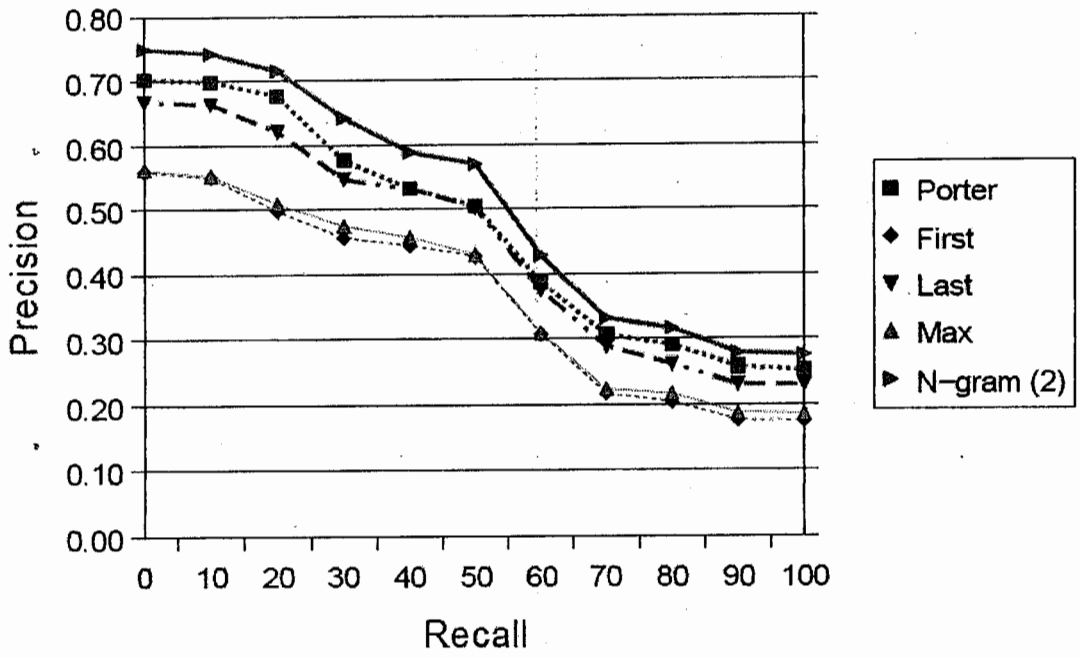


Figure 3.11: N-Grams vs. Other Stemming Approaches for English Language[10]

But the same is unfortunately not true for Arabic. Experiments performed by Mustafa and Radiedeh (2004), showed pure N-Grams matching was not a good option for Arabic due to what they noted was high interference by affixes.

3.2.1 Combining Light Stemming with N-Gram String Matching

Due to the high incidence of infixes as noted by De Roeck et al (2000) and Mustafa (2005), results could be markedly improved by first stemming the word and then computing the string similarities. The affixes used by these researchers were not specified in their papers. We decided to use the light stemmer developed by Leah Larkey, known as the light10. It is now a well reputed stemmer for stemming Arabic text being the best performer at TREC. The following are the list of prefixes and affixes dropped by light10:

Prefixes: ال، وال، بال، كال، فال، لل، و

Suffixes: ها، ان، ون، ين، يه، ية، ه، ة، ي

[10] From Kosinov (2001)

As noted in the prefix list, it consists mostly of removal of definite articles with the exception of waw (و); the prefixes are just the definite article Al (ال) complemented with a preposition or conjunction. The suffix list has a variety of different form of frequently occurring ending affixes which pluralize or feminize the word.

3.2.2 Size of the Substring

The size of the slice of a word to form word substring as gram token has been experimented by researchers. Numerous sizes for N have been proposed, from uni-grams to tri-grams. The various formations are shown in table 3.2 below.

Language	Word	Uni-Grams	Bi-Grams	Tri-Grams
English	Text	T e x t (4)	Te ex xt (3)	Tex ext (2)
Arabic	اسلام	ا س ل ا م (5)	اس سل لا ام(4)	اسل سلا لام(3)

Table 3.2: N-Gram Sizes Compared

With unigrams, in the calculation of SC the order of the letters is ignored. Hence any two letters with common alphabets but different ordering of letters will give a high SC value. This can give undesirable results especially in the case of Arabic which has short words usually derived from a three letter root. A change in the order of the root letters would completely change the meaning of the word. Hence usually, uni-grams are not used in calculating word similarities.

The study for N-Grams size for Arabic by Mustafa et al (2004) shows the bi-grams are better than tri-grams. In their results, as depicted in table 3.3, better performance was obtained for bi-gram retrieval than for tri-grams. This is perhaps due to the shorter word lengths of Arabic word resulting in fewer tri-gram token being created.

	Digram		Trigram	
N-gram method	Total	Avg.	Total	Avg.
Variants retrieved	429	8.58	428	8.56
Relevant retrieved	283	5.66	200	4.0

Table 3.3: Di-Grams (Bi-Grams) vs. Tri-Grams

This result was also obtained by De Roeck et al (2000) who investigated "empirically" that bi-grams are better than "tri-grams."

Consequently, as a result of this researcher, we chose to only investigate bi-grams string matching for retrieval.

3.2.3 Formula for Similarity Co-efficient Calculation

The value of a measure of association increases as the number or proportion of shared n-grams increases. Numerous coefficients of association have been described in the literature. Several authors have pointed out that the difference in retrieval performance achieved by different measures of association is insignificant, providing that these are appropriately normalized. Intuitively, this is what one would expect since most measures incorporate the same information i.e. the number of common unique n-gram token and the total number of unique n-gram tokens in both words. A simple measure of association is

$|X \cap Y|$ *(Simple Coefficient Measure)*

which is the number of shared index terms. This coefficient does not take into account the sizes of X and Y. One of the most popular measures used in document retrieval which takes into account the sizes of X and Y is the Dice Co-efficient:

$\frac{2 * |X \cap Y|}{|X| + |Y|}$ *(Dice's Coefficient Measure)*

This co-efficient is a normalized version of the simple co-efficient measure. As pointed out by van Rijsbergen (1979), failure to normalize leads to counter intuitive results as the following example shows:

Let $S_1 (X, Y) = |X \cap Y|$,

$S_2 (X, Y) = \frac{2*|X \cap Y|}{|X| + |Y|}$

if $|X_1| = 1$ $|Y_1| = 1$ then $|X_1 \cap Y_1| = 1 \Rightarrow S_1 = 1$ and $S_2 = 1$

also if $|X_2| = 10$ $|Y_2| = 10$ then $|X_2 \cap Y_2| = 1 \Rightarrow S_1 = 1$ and $S_2 = 1/10$

$S_1 (X_1, Y_1) = S_1 (X_2, Y_2)$ which is clearly absurd since X_1 and Y_1 are identical representatives whereas X_2 and Y_2 are radically different. The normalization for S_2, scales it between 0 and 1, maximum similarity being indicated by 1.

As De Roeck et al (2000) argued Dice's equation boosts the importance of unique shared substrings between word pairs, by doubling their evidence. Since Arabic words tend to be short, the relative impact of shared substrings will already be dramatic. They replaced the Dice metric with the Jaccard formula to reduce this effect:

$\frac{|X \cap Y|}{|X \cup Y|}$ *(Jaccard's Coefficient Measure)*

An example highlighting this difference is shown in table 3.4.

String	**Bi-Grams**	**Unique Bi-Grams**	**Shared Bi-Gram**	**SC (Dice)**	**SC (Jaccard)**
اسلام	اس سل لا ام	(4) اس سل لا ام	(3) سل لا ام	2*(3)/(4+3) = 0.86	3/((4+3)-3) = 0.75
سلام	سل لا ام	(3)سل لا ام			

Table 3.4: Dice's vs. Jaccard's Coefficient

As seen from the table, Dice's Measure tends to give a higher SC value as compared to Jaccard's. This increase in the case Dice's measure is due to the doubling of the value of number of common N-Grams in two word string tokens.

Various researchers have used different similarity measures so in our study we experimented with both the Dice's measure as well as the Jaccard's measure to see the effect of each on retrieval.

3.2.4 Boundary Space

De Roeck et al (2000) also noted that N-gram size can curtail the significance of word boundary letters (Robertson and Willet (1992). Whereas the non boundary letters appear in two gram tokens, the boundary letters appear in only one hence decreasing their contribution to the SC value calculation. This is especially important since after stemming the word, the remaining portion of the word's boundary letters need to be given their due consideration. To give them opportunity to contribute fully to the SC value, word boundary blanks were introduced (Harman 1991). Also, the larger the number of n-gram tokens, the greater its capacity to mask the shorter substring which can contain important evidence of similarity between word pairs (Adamson and Boreham 1974). The following example shows the effect of introducing boundary blanks n-grams (indicated by a *):

String	Bi-Grams	Unique Bi-Grams	Shared Bi-Gram	SC (Dice)	SC (Jaccard)
اسلام	*ا اس سل لا ام م*	(٦) *ا اس سل لا ام م*	سل لا ام *م (٤)	2*(٤)/(٦+٥) = 0.73	٤/((٦+٥)-٤) = 0.57
سلام	*س سل لا ام م*	(٥) *س سل لا ام م*			

Table 3.5: The Effect of Boundary Blanks

This shows what a great impact boundary space has on retrieval. The SC has fallen considerably between the two words using boundary space. The difference in the two words has expounded due to the inclusion of the boundary space.

3.2.5 Character Contiguity And Differential Weighting

Till now we have seen a string sliced into n-gram token of contiguous characters. However as appears in the literature the term N-Gram can be made up of any co-occurring set of characters in the string e.g. a token made up of the first and third character or the first and fourth character etc. Numerous studies have been carried out with using non adjacent characters in n-grams giving a positive result.

So what is the rationale of having non-contiguous characters for Arabic?

Arabic Morphology involves a complex infix structure. Word infixes may occur in two places: after the first root radical and before the last root radical. This might lead to an assumption that adjacent N-grams might fail to capture the similarity between related infixed words. If this is a valid assumption then we would expect that non-contiguous N-grams with a single character gap should improve the overall performance by lowering the weightage of tokens or eliminating the tokens containing infixes. The non-contiguous bi-grams are formed from the conventional contiguous boundary space bi-grams. Given a character token W (i.e. a string of letters W[1], w[2], ...W[n] over a given natural language alphabet), its set of bi-grams D_w is generated in the following manner:

Contiguous Bi-grams: $D_w = (W_1+W_2, W_2+ W_3,... W_{n-1}+ W_n)$

Non-Contiguous Bi-Grams: $D_w = (W_1+W_2, W_1+ W_3, W_2+ W_3, W_2+ W_4, ... W_{n-2}+ W_n, W_{n-1}+ W_n)$

Let us see an example of the application of this formation given in table 3.6.

Word	Continuous Bi-grams	Non-Contiguous Bi-Grams
سلام	*س سل لا ام م* (5)	*س *ل سل سا لا لم ام ا* م* (9)

Table 3.6: Contiguous and Non-Contiguous

Differential weighting schemes: The objective is to have an algorithm for conflation which gives precedence to root consonants rather than affixes (De Roeck 2000). Light stemming removes some of the affixes. The detection of affixes is not fool proof since letters are common to both affixes and root consonants. However, since affixes are a

closed class, it is possible to identify "suspect" affixes (De Roeck 2000). A list of possible "suspect" affixes used to form different word patterns is given below:

س، أ، ل، ت، م، و، ن، ي، ه، ا

Following De Roeck (2000) we explored the idea of assigning differential weights to n-gram tokens. In their scheme equal weights of 1 was given to all substrings equating the weight contributed by all letters, whether they are root consonants or not. Suspect affixes, however, were not allowed to affect the SC between words on a par with characters contributing stronger evidence. They determined "empirically" that 0.25 weight for strings containing weak letters (و،أ، ي), and 0.50 for strings containing suspect non-weak letter affixes gave the best SC for their datasets.

We implemented a slightly different scheme for differential scheme weighting than De Roeck. Noting that the suspect affixes, م، ت، أ، ي occur only in the beginning we assigned the substring tokens occurring with these beginning letters a weight of 0.5. Similarly, ت، ا occur only at the end of a word, we assigned the token containing these end letters a weight of 0.5. Weak letters occur as infixes after the first letter or before the last letter, as stated earlier; n-grams containing these weak letters were assigned a weight of 0.25. This scheme differs from De Roeck's in that it is more specific as to when to assign the differential weighting to the suspect and weak affixes. This scheme as compared to De Roeck's scheme is less vulnerable to more errors with fewer chances of assigning low weightage to N-Grams tokens with suspect affix letters occurring as root consonants.

String	Non Contiguous Unique Bi-Grams	Shared Bi-Gram	SC (Dice)
عمل	مل(١) عل(١) عم(١) م*(١) ع*(١) *ل(١) *م(١) Total: 7	مل(١) عم(١) ع*(١) *ل(١) *م(١) Total: 5	(2*5) ÷ (7+٦.50) = 0.74
معامل	ما(٠,٢٥) مع (٠,٥) ع*(١) م*(٠) ال(٠,٢٥) ام(٠,٢٥) عم(١) عا(٠,٢٥) *ل(١) *م(١) مل(١) Total: ٦.٥٠		

Table 3.7: Differentially Weighted N-Grams

String	Non Contiguous Unique Bi-Grams	Shared Bi-Gram	SC (Dice)
عمل	مل(١) عل(١) عم(١) م*(١) ع*(١) *ل(١) *م(١) Total: 7	عم(١) م*(١) ع*(١) مل(١) *م(١) *ل(١) Total: ٦	(2*٦) ÷ (7+١١) = 0.٦٦
معامل	عا(١) ما(١) مع (١) ع*(١) م*(١) عم(١) ام(١) ال(١) مل(١) *م(١) *ل(١) Total: ١١		

Table 3.8: Non-Differentially Weighted N-Grams

An example showing this differential weight assignment is given in table 3.7 (on the preceding page). The same calculation without using differential weighting is given in table 3.8.

As the examples show the words عمل and معامل are related but due to affixes their SC value is lower without using differential weighting. Assigning lower weights to the n-grams with the leading م and the infix ا decreases their contribution to the SC value thus giving a higher measure of relatedness between the two words as desired.

3.2.6 Threshold of Relatedness

The two measures of association, Dice's co-efficient and Jaccard's co-efficient both give significantly different values for threshold. Dice's measure gives a comparatively higher value than Jaccard's measure. A suitable threshold level must be chosen for each measure. Choosing a very low level of threshold will result in more expanded words with less but less similar words being clustered together. Likewise choosing a very high value will result in too few related words being clustered together with closely related words getting left out. An example illustrating the conflation class at different levels of threshold using Dice's Coefficient for the word اسلام is shown in table 3.9.

Threshold	Word Conflation Class
0.75	اسلام لاسلام باسلام كاسلام اسلامي اسلاما سلام سلاس سلاسل استسلام اسلاميا اسلاميت لاستسلام اسلامنا اسلامهم اسلاميو اسلاميك بسلام سلامت لاسلو سلامو لاسلاف اسلاد بسلاسل لسلام لاسلم لاسام اسلاف لاسلح سلاما اسلاك لاسلك
0.80	اسلام لاسلام باسلام كاسلام اسلامي اسلاما سلام سلاس سلاسل استسلام اسلاميا اسلاميت لاستسلام اسلامنا اسلامهم اسلاميو اسلاميك
0.85	اسلام لاسلام باسلام كاسلام اسلامي اسلاما سلام سلاس سلاسل
0.90	اسلام لاسلام

Table 3.9: Conflation class at Different Levels of Threshold

So how to choose the correct level of the threshold?

We have experimented with several different levels of thresholds. Starting with a specific minimum value we kept on increasing the threshold value in fixed value interval until the difference in performance was insignificant i.e. the mean average precision value difference was less than 0.01.

Also important is the interval of increase of the SC cut-off. Dice's measure is more sensitive to the increase of the SC value than Jaccard's. Slight change in the threshold value of SC resulted in significant word class change. We experimented with 0.1 interval differences for Jaccard's measure starting with the minimum value of 0.5 for the threshold and 0.05 interval differences for Dice's measure with the minimum starting threshold cut-off at 0.75.

3.3 Query Expansion

Query Expansion basically refers to adding of search terms to a user's query. It is the process of a search engine adding search terms to a user's weighted/un-weighted search with the aim to improve precision and/or recall. For example a search for "car" may be expanded to: car, cars, auto, autos, automobile, and automobiles.

There are two main issues to be considered while adding terms to expand the query: (1) Which terms to include (2) Should the terms be weighted differentially and on what basis to apply the weights terms. Other considerations include whether to expand based on individual terms of the initial query or to expand based on the concept of the query topic. The additional terms may be generated *manually* or *automatically*. Manual expansion involves the human extending the query based on certain heuristic; whereas, in automatic expansion the system automatically expands the query. Automatic expansion may be based on search results referred to as *relevance feedback* or based on a knowledge structure which may be collection dependant or independent. This categorization is shown in figure 3.12.

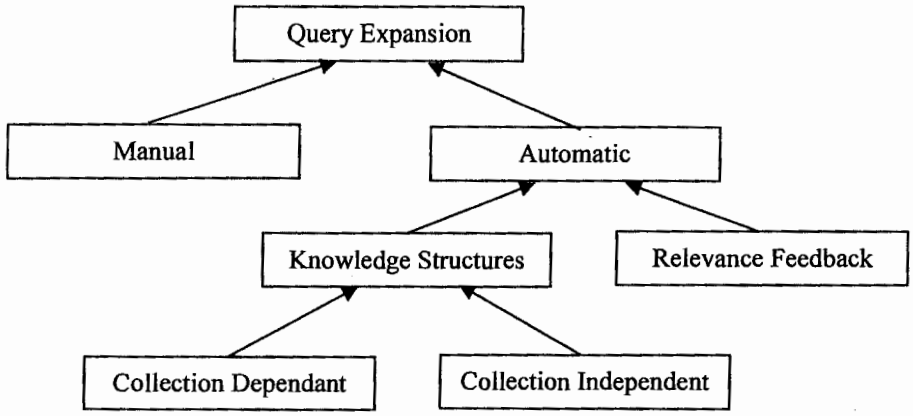


Figure 3.12: Query Expansion Types

The idea behind relevance feedback is to take the results that are initially returned from a given query and to use information about whether or not those results are relevant to perform a new query. Relevance information is utilized by using the contents of the relevant documents to either adjust the weights of terms in the original query, or by using those contents to add words to the query. Relevance feedback is often implemented using the Rocchio algorithm.

Our technique of string similarity based exapnsion falls under the category of automatic expansion based on knowleadge structure making use of the collection dataset. Each term in the query is matched with all the words in the corpus collection in order to

automatically determine all related words for each query term. Hence in this scheme each term of the query rather than the concept of the whole query topic is considered for expansion. Once we have the weighted graph of terms and their related words in the corpus along with a value of the similarity weight there are numerous techniques to expand the query:

Term Counts greater than two: Only those words/stems are chosen for expansion whose word count in the entire dataset is greater than 2. This eliminates words from the corpus which are spelling mistakes or are too uncommon to appear only once or twice. We saw such misspelled word occurrences were very common in the expanded query and believed they had to be removed else they reduce the importance of other words in the expanded query.

Normalization: Each stemmed word in the original query, referred to as a *concept*, can have different number of expanded words. Therefore each concept must have a total weight of 1. Hence all the words under a concept are *normalized* to 1 i.e. weights are assigned to each word of the concept such that the sum is equal to 1. This normalization was done so that the query would not be re-balanced to give more weight to a concept merely because it had many synonyms.

Formula: Weight of term = (1/number of terms in each concept)

Normalized Weights of Each Query Term Based on SC Values: We considered assigning weights to each of the terms of a concept that were expanded. Thus a weight was assigned based on the SC values of the words that are above the chosen threshold. Highest weight was given to terms in a concept which had an SC value of 1 and the sum of all the weights in a concept was equal to 1 as required by normalization. Formula: Weight of term = (SC Value of term/Sum of all SC term values of a concept)

Normalized Weights of Each Query Term Based on Frequency Counts (FC): Weight are assigned to words of a concept depending on the frequency of occurrence of the word in the dataset and then normalized. The words were assigned the relative weights based on

their term count in the corpus. In this way words which occur more frequently were given emphasis over words which occur less frequently in the document collection. Formula: Weight of term = (FC Value of term/Sum of all FC term values of a concept)

4. Results and Evaluation

4. Results and Evaluation

This chapter presents the results of numerous evaluation experiments conducted to achieve the required objectives outlined in the problem statement along with an analysis of the experimental results. We will briefly outline the initial experimental setup followed by the data analysis and discussion of the results.

4.1 Initial Setup

This section describes the initial steps followed to prepare the data, indexes and the query set for the experiments.

- The data set described in section 3.1.1 consists of 383,872 documents. The files, which were originally encoded with Unicode in UTF-8 format, were converted to Windows Code Page 1256 encoding (CP 1256) because the Lemur Toolkit can only handle Arabic language in (CP 1256) encoding.
- The same step was followed for the query set; the queries were converted from the ASMO 708 encoding to (CP 1256) encoding.
- Title for each of the 25 queries was extracted from the original query set to be used for the study.
- Two indexes were built; one using the light 10 stemmer and the other using only the table Normalization and stopword removal functions in the Light-10 stemmer. The normalization and stopword removal that are carried out has already been discussed in section 3.1.3.
- In order to avoid confusion each technique was given a code that was used to represent it throughout the experiments:
    - No stemming: NoST
    - N-Gram procedure: NG
    - N-Gram procedure using Dice's measure: DIC
    - N-Gram procedure using Jaccard's measure: JAC

- Simple Adamson's N-Gram Procedure: SMP
- N-Gram Procedure with Boundary Grams: BND
- N-Gram Procedure with Non Contiguous Grams: NCT
- N-Gram Procedure with Differential weighted Grams: DIF
- N-Gram Procedure with a specific threshold e.g. 0.75
- Light-10 Stemmer: L10
- Simply Expanded Query: EXP
- Expanded Query with Term Count greater than 2: GT2
- Expanded Query with Normalization: NRM
- Expanded Query with Weighting according to Sim Coeff: WSC
- Expanded Query with Weighting according to Term Count: WTC

Techniques for N-Gram procedure are sometimes represented by combinations of codes to emphasize the particular property of the procedure being compared. For instance, when comparing N-Gram parameter combinations, L10_NG_DIC_BND represents N-Gram procedure involving light stemming and boundary N-Grams using Dice's measure for similarity calculation.

4.2 Results, Data Analysis and Discussion

In this section we present the results of the experiments conducted. We first start by presenting the results for highlighting the importance of using the hybrid N-Gram based stemming approach. Thereafter the results for different parameters for N-Grams using the two measures of association, Dice and Jaccard, are evaluated. The best threshold level is also obtained empirically. Finally a comparison of the best hybrid N-Gram procedure approach is compared to simple and straight-forward light stemming approach using the industry recognized light-10 stemmer.

4.2.1 No Stemming and Hybrid Stemming

We first start by comparing search based on index of document without any stemming (normalized and stopwords removed) to N-Gram based Stemming and hybrid N-Gram Stemming (which includes N-Gram procedure as well as light stemming):

Experimental Runs:

- Raw, non-stemmed retrieval: NoST
- Raw, non-stemmed N-Gram based retrieval: NoST_NG
- Stemmed(light10) N-Gram based retrieval: L10_NG

Comparison is shown in the figure 4.1.

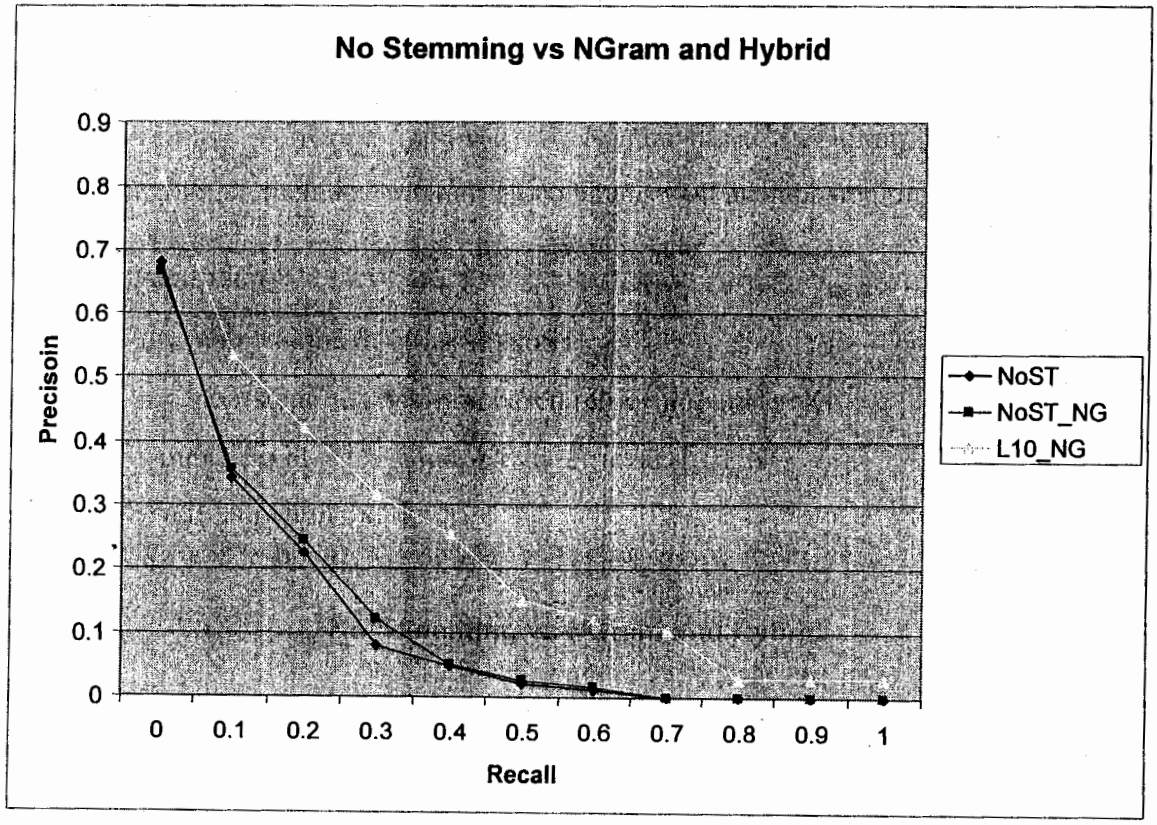


Figure 4.1 : Comparison of N-Grams And Hybrid N-Grams Against Unstemmed Document Retrieval

Results Summary:

Run ID	Rank*	P-Value**	Significance***
NoST	0		
NoST_NG	1	0.0526	Slightly Insignificant
L10_NG	2	< 0.0001	Highly Significant

** Higher rank shows better performance*

*** Relative to NoST.*

**** A p_{τ}value < 0.05 shows significant improvement*

Analysis of Results:

- Without using Stemming, the use of N-Grams for conflation, showed improvement but slightly insignificant.
- But, when we applied light10 stemmer before N-Grams conflation (Hybrid Stemming) the improvement is highly significant

Discussion:

Previous researchers (Mustafa and Radaideh, 2004) working on non-standard TREC Dataset showed no significant improvement in results using N-Gram procedure; so we first showed the comparison of plain non stemmed index retrieval (NoST) to an expanded query using simple (Adamson's) N-Gram procedure (NoST_ NG) as seen in figure 4.1. This result tallies with the previous work showing pure N-Gram based stemming to be not very fruitful.

Also, as De Roeck and Mustafa determined in their experiments that combining light stemming with N-Gram procedure (Hybrid Stemming) showed considerable improvement in results. Hence, our experiments with Hybrid Stemming (L10_NG) show considerable improvements over unstemmed non N-Gram based retrieval (No_ST).

So far these results are in accordance with the results of past research on wordlist. This was our motivation behind adopting and further researching this particular technique for stemming. The results have been confirmed using a large-sized standard TREC Corpus

which was not available at the time of research of the studies carried out by these past researchers.

Before we go on to investigate the various parameters for N-Gram Procedure we will first see how we expanded our query in order to achieve the best results.

4.2.2 Query Expansion

In section 3.3, we discussed several methods of expanding the query which incorporated all the word variants of a word. We experimented with all the different methods outlined in the section.

Experimental Runs:

- All words in the conflation class expanded: EXP
- Only terms with Term Count (TC) greater than 2 in dataset expanded: EXP_GT2
- TC > 2 and Normalized: EXP_GT2_NRM
- TC > 2, Normalized and weighted according to SC: EXP_GT2_NRM_WSC
- TC > 2, Normalized and weighted according to TC: EXP_GT2_NRM_WTC

Comparison is shown in the figure 4.2.

Results Summary:

Run ID	Rank	P-Value*	Significance
EXP	0		
EXP_GT2	1	--	Highly Insignificant
EXP_GT2_NRM	2	0.484	Insignificant
EXP_GT2_NRM_WSC	3	0.2061	Insignificant
EXP_GT2_NRM_WTC	4	0.0003	Highly Significant

** Relative to EXP.*

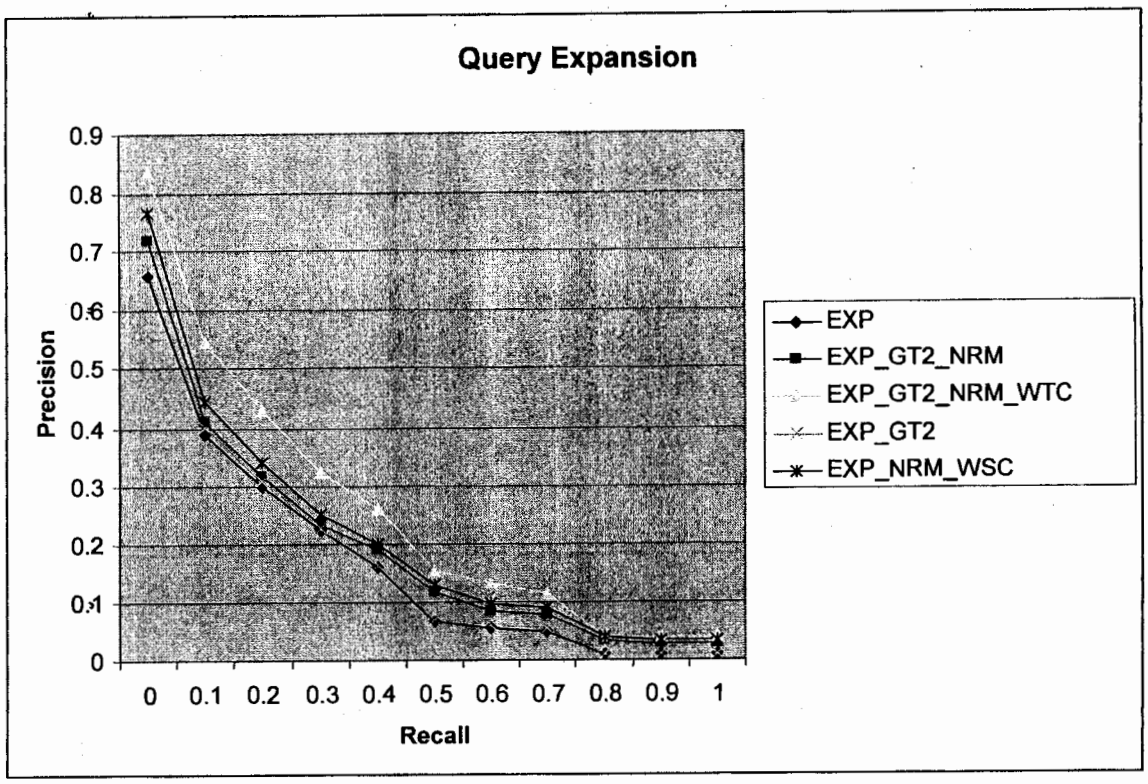


Figure 4.2 : Comparison of Various Query Expansion Methods

Analysis of Data:

- No improvement could be seen by excluding words whose count is less than 2 in the corpus as we had thought.
- As expected normalization did help to improve results but not significantly
- Incorporating weights showed marked improvements:
  - With SC weighting the result showed little improvement
  - But with TC weighting the results are much better

Discussion:

These results show that normalization and appropriate weighting is of critical importance. Using only normalization gave equal weightings to all the words of a concept which does not take into consideration the relative importance (usage) of the word in the dataset. The best result obtained using normalization and weighting based on Term Count shows that there are varied term frequency counts in the corpus literature some having very small

value while others very large; hence the weight assigned to the words according to the term counts helped to boost the results remarkably. This differential weighting of terms makes sense as all words in an equivalence class are considered to be the same therefore the different weights assigned only helps to give each word the appropriate importance as found in the corpus. If same weight is assigned to a word which occurs less frequently as compared to one which occurs more frequently then documents containing the less frequently occurring word would tend to be scored higher by the retrieval scoring function thus adversely affect performance.

Next, we investigated the different parameters of N-Grams described in section 3.2 using the best query expansion strategy just explained. These various parameter were compared under the two schemes of similarity association measures, the Dice's and the Jaccard's measure.

4.2.3 N-Gram Parameters Using Dice's Co-efficient

Using the original measure of association in Adamson's N-Gram procedure (1994), the Dice's Coefficient, we first determined a suitable cut-off level of threshold to be used. Thereafter this cut-off value was used in subsequent experiments with N-Gram parameters.

4.2.3.1 Choosing an SC Cut-off Value

Starting at threshold cut-off value of 0.75 and increasing the SC cut-off with small interval increments of 0.05, we want to find a suitable cut-off value at which the improvement by a certain increase is significant and subsequent increments are insignificant.

Experimental Runs:

- Simple Hybrid N-Grams using Dice's measure with SC cut-off at 0.75: NG_DIC_SMP_075
- At cut-off 0.80: NG_DIC_SMP_080

- At cut-off 0.85: NG_DIC_SMP_085
- At cut-off 0.90: NG_DIC_SMP_090

Comparisons are shown in figure 4.3.

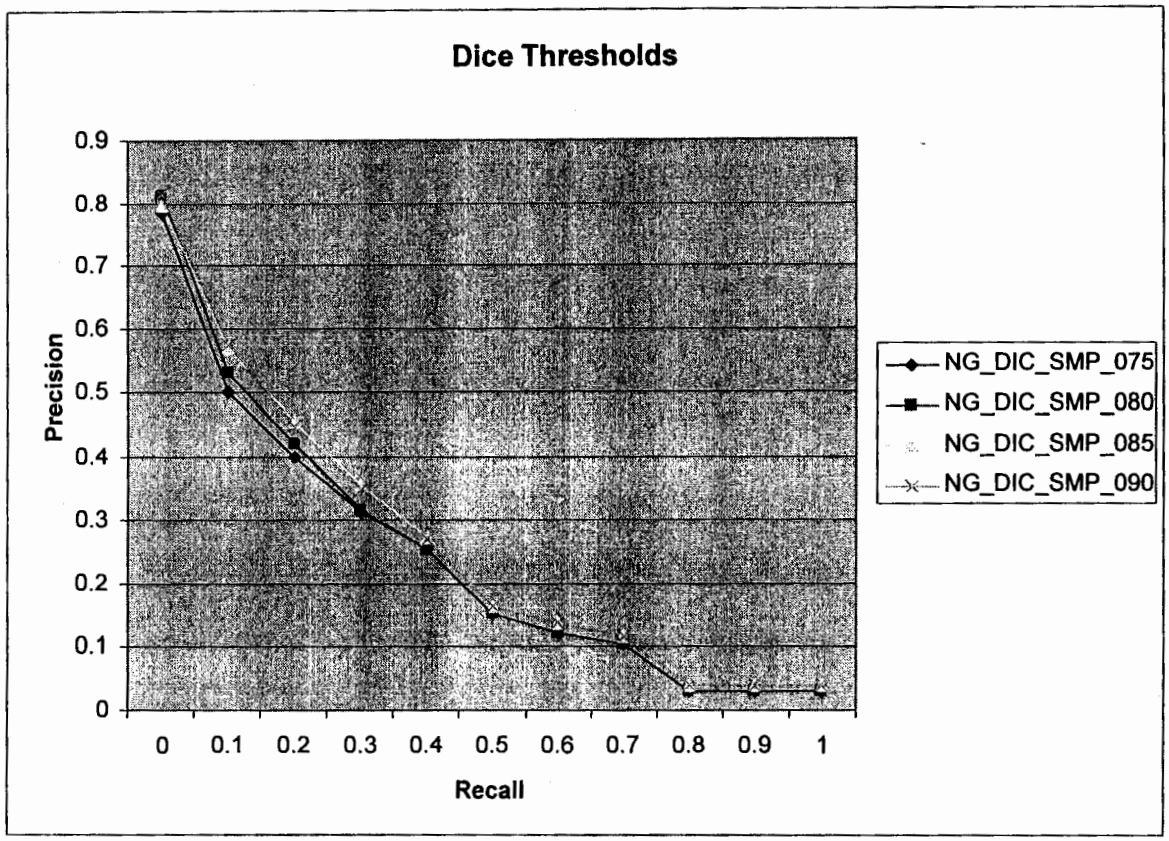


Figure 4.3 : Comparison of Different Threshold Levels for Dice's Coefficient

Result Summary:

Run ID	Rank	P-Value*		Significance
NG_DIC_SMP_075	0			
NG_DIC_SMP_080	1	0.171		Insignificant
NG_DIC_SMP_085	3	0.036		Significant
NG_DIC_SMP_090	2	0.0526	0.337**	Insignificant

** Relative to NG_DIC_SMP_075*

*** Relative to NG_DIC_SMP_085*

Analysis of Results:

- No significant improvement is seen at threshold level 0.80
- Best threshold level is 0.85 for Dice's Measure with the improvement being significant.
- At a threshold of 0.90 the improvement relative to the 0.85 level is highly insignificant hence makes it an appropriate value for SC cut-off.

4.2.3.2 Comparing N-Gram Parameters

Using the threshold level 0.85 we will now give a comparison of the three enhancements made to the simple N-Grams procedure discussed in section 3.2 namely, boundary N-Grams, Non-Contiguous N-Grams and differential N-Grams.

Experimental Runs:

- Hybrid Simple N-Grams using Dice's measure: NG_DIC_SMP
- Boundary N-Gram: NG_DIC_BND
- Non-Contiguous N-Grams: NG_DIC_NCT
- Differentially weighted N-Grams: NG_DIC_DIF

Comparisons are shown in figure 4.4.

Summary of Results:

Run ID	Rank	P-Value*	Significance
NG_DIC_SMP	3		
NG_DIC_BND	1	0.095	Insignificant
NG_DIC_NCT	2	0.305	Insignificant
NG_DIC_DIF	0	0.076	Insignificant

** Relative to NG_DIC_SMP*

Analysis of Results:

- Simple N-Gram procedure turned out to be the best in comparison to all the enhancements done to the technique, though the difference is insignificant
- Worst result is seen in the case of differential weighting, although there too the p-value shows it to be insignificantly worse.

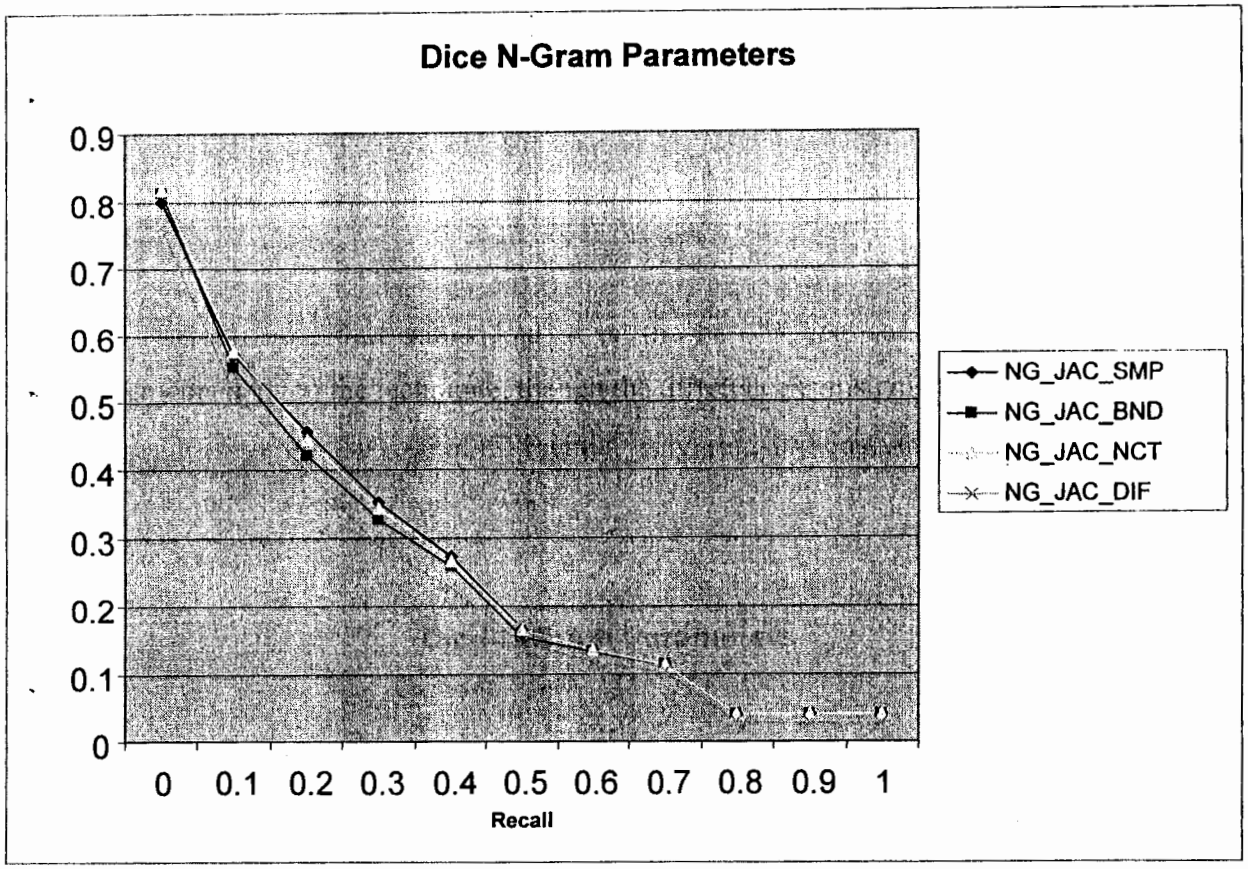


Figure 4.4: Comparison of N-Gram Parameters using Dice's Coefficient

Discussion:

Differentially weighted N-Grams were used by De-Roeck (2000) who got good results for her data. Using a slightly more refined version of De-Roeck's differential weighting scheme, the results we have obtained on TREC Dataset are not so encouraging. In order to possibly see the failure we looked closely at the expanded query of the differentially weighted N-Gram conflation techniques which can be seen in table 4.1.

Original	فنون العرض و المؤسسات الاسلامية في العالم العربي					
Stemmed	عرب	عالم	اسلام	مؤسس	عرض	فن
Expanded (Differential-Dice)	عرب	مالام	سالام	مؤسس	عرض	فن
	اعرب	عالم	اسلام	مؤس	تعرض	
	تعرب	الم	باسلام	مؤسست	يعرض	
	عربت	معالم	كاسلام	مؤسسا	عرضت	
	يعرب	اعلم	سالما	لمؤسس	معرض	
	عربا		اسلاما	بمؤسس	عرضا	
	معرب		لاسلام	كمؤسس	اعرض	

Table 4.1: Expanded Queries of Simple and Differential Weighted N-Grams Procedure Compared

Interestingly, the procedure for differential weighted N-Grams correctly conflates the word variants of a word in the corpus almost exactly in accordance with the rules of the Arabic grammar for word formation as defined in the procedure for differentially weighted N-Grams; but unfortunately the words do not relate semantically very well to the word used in the particular context in the query. For example the words,تعرب, اعرب, يعرب , are variants of a verb which means *to parse*, having no relation to the word عرب, meaning *the Arab World*. Similar is the case for the word, عرض. Although the variants of the word مؤسس are correctly formed, it seems that incorrect forms highly dominate over the correct forms in the expanded query hence giving us the undesired results.

4.2.4 N-Gram Parameters Using Jaccard's Coefficient

Like in the case of Dice' Coefficient, we first determined a suitable cut-off level of threshold to be used. Thereafter using that SC cut-off value we did the comparison of the different N-Gram parameters.

4.2.4.1 Choosing an SC Cut-off Value

With Jaccard the starting threshold cut-off value of 0.50 with interval increments of 0.1 were used. Again our aim is to find a suitable cut-off value at which the improvement by a certain increase is significant and subsequent increments are insignificant.

Experimental Runs:

- Boundary Hybrid N-Grams using Jaccard's measure with SC cut-off at 0.50: NG_JAC_BND_050
- At cut-off 0.60: NG_ JAC _BND_060
- At cut-off 0.70: NG_ JAC _BND_070

Comparisons are shown in figure 4.5.

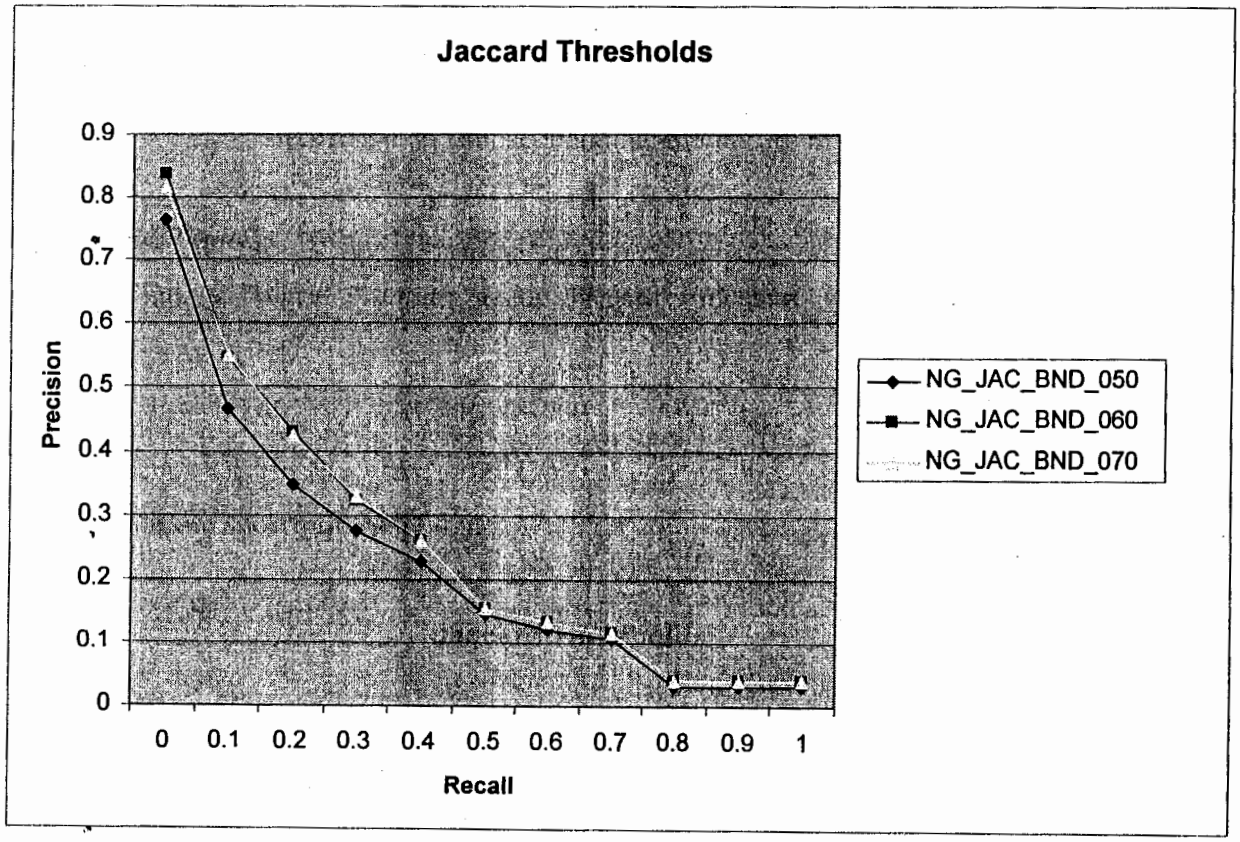


Figure 4.5: Comparison of Different Threshold Levels for Jaccard's Coefficient

Summary of Results:

Run ID	Rank	P-Value	Significance
NG_JAC_BND_050	0		
NG_JAC_ BND _060	1	0.017*	Significant
NG_JAC_ BND _070	2	------**	Highly Insignificant

** Relative to NG_DIC_SMP_050*

*** Relative to NG_DIC_SMP_060*

Analysis of Results:

- Significant improvement is seen at the cut-off level of 0.60
- No further improvement was obtained at the level of 0.70 and above.

4.2.4.2 Comparing N-Gram Parameters

Here we will illustrate the results of only one of the three varied N-Gram parameters over the simple N-Gram procedure namely, Boundary. We chose to ignore differential weighting and hence non contiguous procedures for N-Grams as it did not turn out to be a promising approach as seen in the case of Dice's Measure N-Gram Parameter comparisons.

Experimental Runs:

- Hybrid Simple N-Grams using Jaccard's measure: NG_JAC_SMP
- Boundary N-Gram: NG_JAC_BND

The comparison is shown in figure 4.6.

Summary of Results:

Run ID	Rank	P-Value*	Significance
NG_JAC_SMP	0		
NG_JAC_BND	1	0.058	Slightly Insignificant

** Relative to NG_JAC_SMP*

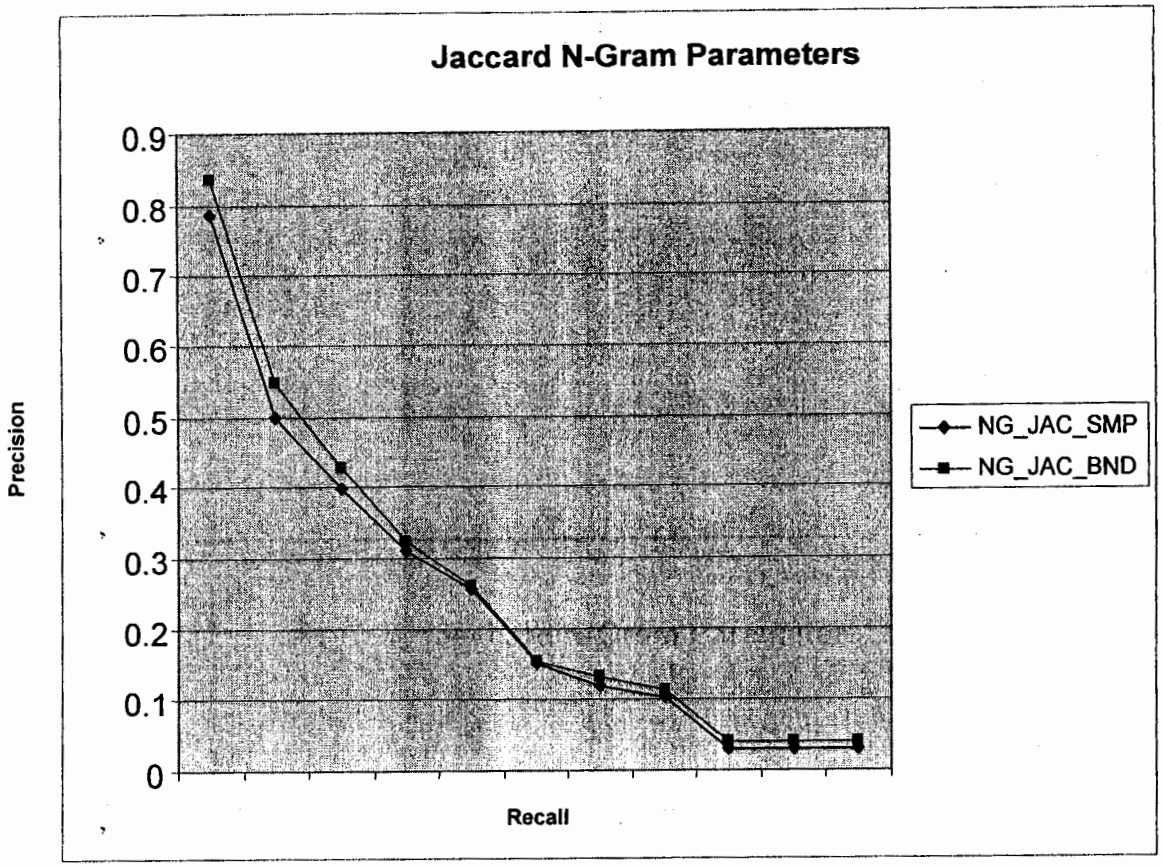


Figure 4.6: Comparison of an N-Gram Parameter using Jaccard's Coefficient

Analysis of Results:

- Using Boundary N-Grams gave slight improvement over the simple N-Grams approach.

Discussion:

The radically different effect of using boundary N-Grams in the case of Dice's and Jaccard's measure is a rather surprising result. Including boundary space tends to lower the SC value of words with dissimilar boundaries. This drop in SC value appears to have a significant impact in the case of Dice's measure, considerably reducing number of words conflated and hence deteriorating performance. Experiments we conducted by lowering the SC cut-off for Dice's measure with the expectation to increase the conflation class size but did not help to achieve better performance.

4.2.5 Dice's vs. Jaccard's Coefficient

Now we wished to compare if using Jaccard's measure would give any advantage, as asserted by De Roeck (2000) (see section 3.2.3), over use of the originally proposed Dice's measure in Adamson's approach.

Experimental Runs:

- Hybrid Simple N-Grams using Dice's Measure: NG_DIC_SMP
- Hybrid Boundary N-Grams using Jaccard's Measure: NG_JAC_BND

A comparison showing the best performers in each approach are compared in figure 4.7.

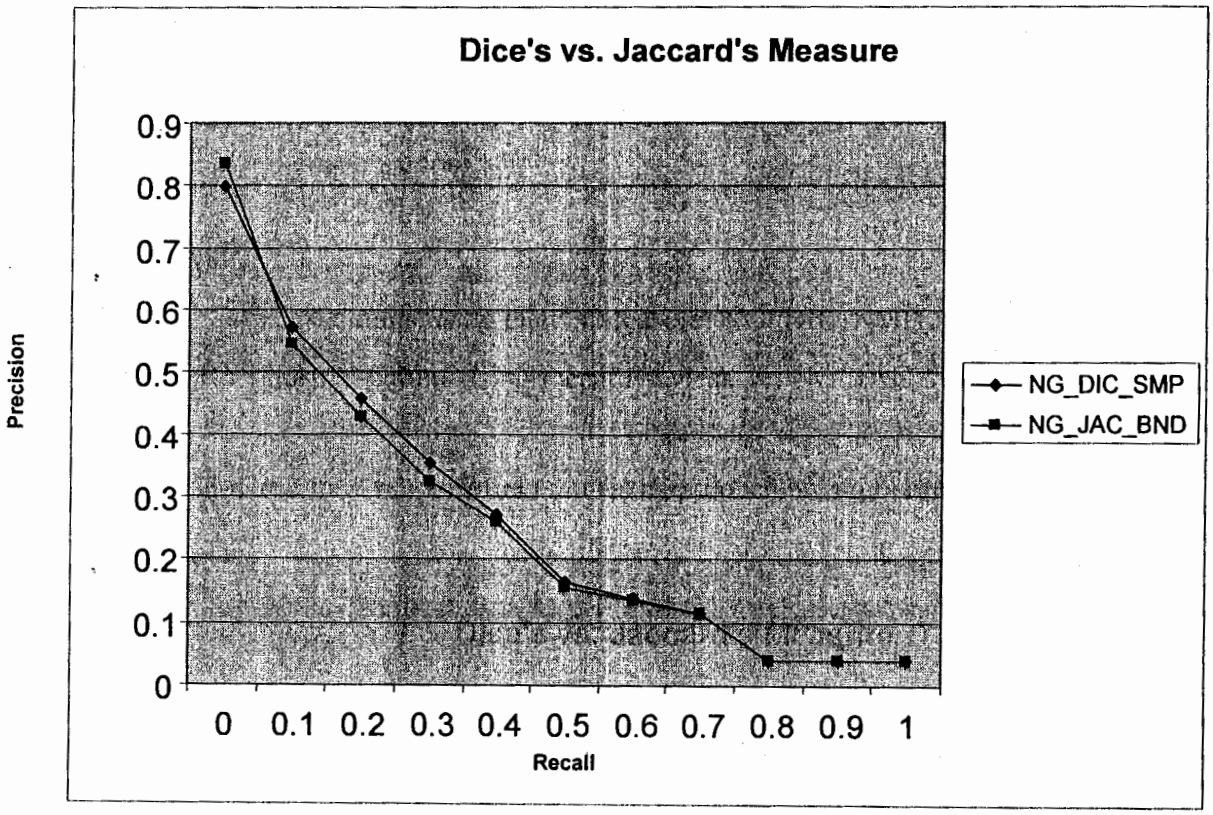


Figure 4.7: Comparison of Dice's and Jaccard's Measure

Summary of Results:

Run ID	Rank	P-Value*	Significance
NG_DIC_SMP	1		
NG_JAC_BND	0	0.233	Insignificant

** Relative to NG_JAC_BND*

Analysis of Result:

- Dice's measure using simple N-Grams shows slightly better performance than Jaccard's Approach

Discussion:

De Roeck's claiming better performance using the Jaccard's measure appears to be invalid. This and previous experiments establish the superiority of the original Adamson's approach using simple N-Grams and Dice's coefficient for association over all other variations proposed in the past by researchers. Despite the added intricacies in the variation of each N-Gram procedure they failed improve retrieval performance.

Now that we have the best hybrid N-Gram procedure for stemming we wish to evaluate its performance against the industry's well-acknowledged light stemmer, the Light-10. This will give us a true idea of how good our hybrid stemming approach really is in comparison to simple light stemming.

4.2.6 Hybrid Stemming vs. the Stemmer, Light 10

Our Hybrid Stemming approach itself incorporates the light stemmer, Light-10. By doing a comparison of Light-10 with the Hybrid N-Gram stemmer we will determine how much better, if at all, is the hybrid approach over and above the light stemmer.

Experimental Runs:

- Light10 stemmer: L10
- Hybrid Stemmer:Light10 + N-Grams(simple, using Dice's measure): L10_NG

The comparison is shown in figure 4.8.

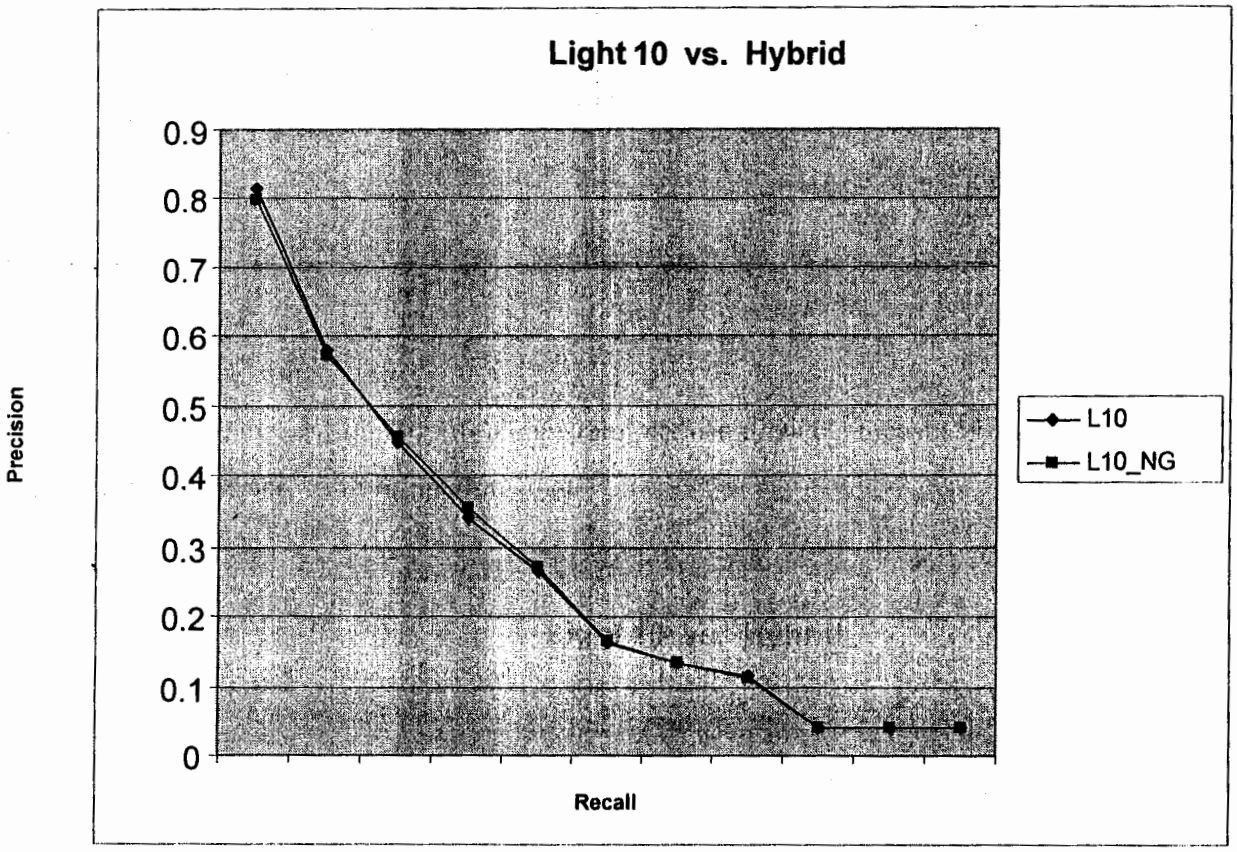


Figure 4.8: Comparison of Hybrid N-Grams against Light Stemmer, Light-10

Summary of Result:

Run ID	Rank	P-Value*	Significance
L10	0		
L10_NG	1	0.245	Insignificant

** Relative to L10*

Analysis of the Result:

- Hybrid stemming appears to be very slightly better than the light-10 stemmer

Discussion:

Here we see a **twist in the tale**: The superior performance of the Hybrid approach that we saw in the figure 4.1 was in fact due to the high quality light 10 stemmer. As the results here show that the combined approach virtually gave no improvement over the retrieval performance of the light-10 stemmer. Since no researcher in the past has really given such a comparison we were deluded by the improvement of the hybrid approach to be actually of the N-Grams procedure whereas in-fact the real role played in the improvement of the hybrid approach was of the underlying light stemming that is incorporated.

Original	فنون العرض و المؤسسات الاسلامية في العالم العربي					
Stemmed	عرب	عالم	اسلام	مؤسس	عرض	فن
Expanded	عرب(٦٢٥٧١)	عالم(٤١٠٨٩) معالم(٢٥٣) عالما(٦٩) لعالم(٣٢) لمعال(٤) بعالم(١٢) كعالم(٤)	اسلام(٥١٦١١) لاسلام(٧٣) باسلام(١٤) كاسلام(٣) اسلامي(١٢) اسلاما(٨) سلام(٦٦٤٨٨) سلاس(٤) سلاسل(٣٤)	مؤسس(٥٨٨٨) مؤسست(١١٣)	عرض(٧٦٢٨)	فن(١٠٦٦٢)

Table 4.2: Hybrid N-Gram Expanded Query with Term Count

A closer analysis of the data reveals the reason of the failure of the Hybrid stemmer. Consider the original, stemmed (using light-10) and expanded (using simple Adamson's N-Gram Procedure) query along with the term counts of each term in the corpus, shown in table 4.2 (on the preceding Page). The table shows the expanded query of each stemmed word from the original query along with each expanded word's term count in the corpus. This expanded query shows that although the N-Gram procedure has done well to identify most of the word variants in the corpus literature which is why past researchers got good results for the hybrid approach, but these word variants are so few in number in comparison to the stemmed words that they hardly showed any improvement in the over all performance. Only one word variant in this query, سلام gives comparable term count to it original stemmed word اسلام but the word, سلام, itself is not directly related to the sought word, اسلام. It seems obvious that light-10 stemmer with its carefully selected list of affixes was able to conflate of most of the word variants occurring in the corpus literature. These few variants that are missed out by light-10 are in any case present in the documents containing the high frequency occurrences of the stemmed conflated words.

4.3 Evaluation

The main reason of the N-Grams technique's failure to show an improvement over the light-10 stemmer is stated as follows: Light-10 is a very robust well tested stemmer which conflates most word variants. Arabic text, specifically newswire, contains so many definite articles that most word conflation is accomplished simply by removing the definite article ال (Al) and its variant found in light-10 suffix list. The conjunction و (waw) and most of the other frequently occurring suffixes which are removed by the light-10, misses out only a few occasionally occurring variant forms of a word. These occasional forms are also likely to occur in the same document as the other forms, so the documents that are retrieved by the light-10 are most likely to contain the missing forms as well, without losing out on performance. It is these occasional forms that are captured by the N-Grams matching technique but unfortunately do not show much improvements

in the overall performance since it retrieves almost the same documents that have been retrieved by Light 10 stemmer.

4.3.1 Original Contribution

In this work we have accomplished the following novel tasks which have not so far been given attention by any researcher:

- Design of an Arabic Information Retrieval system based on Hybrid N-Gram for conflation of words.
- Evaluation of the Hybrid Technique using standard TREC Dataset.
- An unprecedented comparison the Hybrid approach against the industry recognized light stemmer, Light-10.
- We successfully used a novel query expansion technique to accommodate conflation classes

In the past, due to absence of standard sizable datasets much work on Arabic IR has been unaccredited. For stemming which is critically important for Arabic it is difficult to show the accuracy of a particular technique unless it is tested on an actual document retrieval system. Use of wordlists, in the past, to evaluate the performance of a stemming technique does not necessarily indicate a good retrieval performance. The words retrieved for a particular technique are judged relevant or irrelevant according to the dictionary meanings of the words retrieved. This is misleading as each word in Arabic, without the use of diacritics, can be a noun or verb having various meaning; in an IR system the documents contain the words in a particular context and therefore the stemmed or conflated word classes must conform to the meaning of the word used in that particular context. The true realization of a particular stemming technique needs to be evaluated on a document retrieval system using a sizable dataset such as the TREC dataset which is exactly what we accomplished in this research. The true standing of the hybrid stemming approach has come to light as a result using a sizable corpus for evaluation.

4.3.2 Impacts of this Research

So far there is no standard stemmer available which is used to stem Arabic. This is in contrast to English for which there exist several standard stemming algorithms such as the Lovin's and the Porter's stemmer. As we have seen and mentioned before that Arabic is a Language which contains many inflections of a word while being a morphologically complex language. Little work that has been done on stemming for Arabic so far has shown to give significant improvements in the performance for retrieval. This research is at a preliminary stage. With very few datasets available for testing and evaluation, the stemmers have so far been tested for a particular dataset and not evaluated as to how they would fair on other types of datasets. This area of work holds lot of promise for Arabic IR.

Our work was to investigate a particular technique for stemming Arabic text. Through this work we have managed to bring to light the weaknesses and strengths of the investigated technique which in the past showed promising results. The technique lacked credibility as it had not been tested on any standard for the world to judge against. Hence, there was a question mark about the usefulness it holds for Arabic IR. Also as a result of this work we again brought to light the supremacy of the Light-10 light stemmer which is gaining recognition in the research community and would probably become an industry standard stemmer for stemming Arabic text.

5. Conclusion and Future Research

5. Conclusion and Future Research

5.1 Accomplishment of Our Objectives

In this research we have accomplished our objectives that we had stated in section 2.6. The three key issues that were catered to in this work, as stated in the objectives, were verified experimentally:

- *Design of an Arabic Information retrieval system to determine Hybrid Stemming Superiority*: The degree of improvement in retrieval performance of pure N-Gram matching based stemming and the Hybrid stemming approach over raw (unstemmed) retrieval.

- *N-Gram Parameter*: The effect of variation in the parameters of N-Gram procedure in the net retrieval over the traditional Adamson's N-Grams approach.

- *Comparison to Light-10*: How well does the Hybrid approach perform against the industry-recognized stemmer, Light-10 by Leah Larkey.

The outcomes of these experiments are summed up as follows:

Hybrid Stemming Superiority: Research in the past showed pure N-Gram approach not to be a good performer for Arabic text. However, as experimenter showed that by performing light stemming before doing N-Gram matching can considerably improve the results of the retrieval. We carried out this experiment on the TREC 2001 dataset and the results confirmed the outcomes of results of past researchers. It showed the pure N-Gram approach to be *insignificantly* better but the hybrid approach to perform *significantly* better than raw unstemmed retrieval.

N-Gram Parameter: We experimented with altering several N-Gram parameters, namely (i) including Boundary Spaces (ii) Character Contiguity (iii) Differential weighting (iv) Measure of Similarity, and (v) Threshold cut-off level. We found that there was insignificant difference in the performance by accommodating each of the parameters with traditional Adamson's approach, involving simple contiguous bi-grams with one character overlap without the use of boundary space using the Dice's coefficient of association, giving over all best performance.

Use of differentially weighted N-Grams based on presence of morphemes thus contaminated the purely statistical nature of N-Gram matching. But results obtained were not encouraging as past researchers have thought it to be. Although using this technique gave us word variants conforming to grammatical rules, but the class tended to more often include unrelated forms of the noun/verb in context.

Comparison to Light-10: Comparing the combined, hybrid techniques (Light-10 and N-grams procedure) gave us an eye opener result. The actual deceptive superiority of the hybrid approach, seen in the first experiment, was in fact attributed to the underlying light stemmer; N-Grams matching hardly played a part in the improvement of the result. Closer analysis of the data revealed the cause of failure of the Hybrid approach. The word variants expanded by this method included valid and relevant words but these variants were so few in number in proportion to the light-10 stemmed words that they received a very low weighting in the expanded query to have significant effect on retrieval performance.

5.2 Conclusion

We have seen that stemming is of key importance having a large effect on Arabic information retrieval, far more than the effect found for most other languages. The results obtained from the first experiment showed the clear difference in the results with the light stemmer light-10 outperforming the unstemmed retrieval. This is consistent with the hypothesis of Popovic and Willett (1992) and Pirkola (2001) that stemming should be particularly effective for languages with more complex morphology.

We have compared light stemming with several different variations of N-Gram stemming approaches some incorporating morphological concoctions: performing lighting stemming before N-Grams; differentially altering N-Gram tokens based on presence of inflectional morphemes. Although the hybrid N-Gram stemming (which incorporates the light stemmer) can equal the light stemmer, we have not been able to attain significantly better performance using the hybrid approach.

Why would the sophisticated hybrid N-Grams procedure not perform show significant improvement over a simple stemmer? The main reason of the N-Grams technique's failure to show an improvement over the light-10 stemmer is that the Light-10 stemmer is a very robust and well tested stemmer which conflates most word variants giving little opportunity to N-Grams technique to improve performance. Similar results were obtained by Larkey et al (2005) while comparing Light-10 to morphological analyzers. There too the simple light-10 performed at par with the sophisticated technique of morphological analysis. They, like our-selves, also noted that it is sufficient for information retrieval that many of the *most frequently* occurring forms of a word be conflated without the need to worry about the less frequently occurring forms.

Although the word variants conflated by the Hybrid stemmer do have a high level of accuracy, it just so happens that the particular corpus on which the evaluation has been carried out contains few occurrences of the word variants in the conflation class. The inefficiency of the Hybrid approach is seen to be so, for the case of Arabic newswire text which contains formal terminology. Perhaps in other less formal document sets one would expect more occurrences of the other word variants to be seen, which the Hybrid technique conflates, in which case we would expect a definite improvement in performance.

N-Grams based string matching has other disadvantages too. There are factors which make this approach unfeasible and impractical:

(i) $O(n^2)$ cost of computing the word similarities of all the unique words in the corpus collection. This is exorbitant given that each corpus may contain over a few hundred thousand words.

(ii) The high storage requirements for keeping track of all similarity values which is again $O(n^2)$.

In the light of the studies in this research we have been able to identify the weaknesses of the N-Grams matching with application to a practical retrieval search and standard evaluation dataset. The superiority of the simple light stemmer, light-10 has once again come to light, remaining unchallenged and being so far the best recognized stemmer for Arabic experimented with at TREC.

5.3 Future Work on Stemming

The need to use stemming for Arabic retrieval has become well established. Even though the Light-10 stemmer has proven to be effective, further comparison and investigations for stemming algorithms in Arabic are needed. Exploring different algorithms as we did in this study may lead to a successful stemmer; either the Light-10 becomes a standard stable algorithm or a better one may be found. These studies are only a beginning. Refinements to build an industry standard stemmer for Arabic as the Porter's or Lovin's Stemmer have become a standard for English language, improvement along the following lines could be explored:

Improving N-Grams: More Intelligent and carefully determined differential weighting schemes could be explored which assign lower weight to less likely affix N-Gram tokens. This technique perhaps if intelligently used still does hold a promise as it can handle both affixes and broken plurals. Developing a stemming algorithm that can better handle the broken plurals in the Arabic language without making too many mistakes could be very useful for retrieval. The N-Grams based stemmers that were used in our study did handle this problem reasonably well but not all word forms were captured. The Arabic language

has a considerable number of these broken plural words, and if there is an efficient algorithm that can reduce them to their singular form, it could improve the retrieval results significantly.

Light Stemming: A more intelligent and sophisticated light stemmer could be explored; an algorithm could be explored which carefully analyzes the pattern of a word before applying a particular kind of affix removal. Arabic text tends to have recurring suffixes which could be removed recursively at each stage. Three letter affixes which are also quite common in Arabic could be appropriately removed by giving consideration to the length of a word. Similarly infixes which tend to form word pattern can also be identified and carefully removed.

Morphological Analyzers: Perhaps a better morphological analyzer and better use of morphological analysis to conflate words, could work better than a light stemmer as studied by Larkey et al (2005). They had only tried a few "obvious" alternatives to improve the performance of a morphological analyzer. Ultimately they claimed that one would like to be able to conflate all the inflected forms of a noun together, including broken plurals, and all the conjugations of a verb using morphological analysis, which at the moment is not possible. Clearly, they believe, there is room for future work that makes intelligent use of morphological analysis in information retrieval.

References

References

[1] Mohanned Momani (2007) and Jamil Faraj, "A Novel Algorithm to Extract Tri-Literal Arabic Roots", Computer Systems and Applications, Institute of Electrical and Electronics Engineering / Arab Computer Society International Conference, pages: 309 - 315.

[2] Hayder K. Al Ameed (2006), Shaikha O. Al Ketbi, Amna A. Al Kaabi, Khadija S. Al Shebli, Naila F. Al Shamsi, Noura H. Al Nuaimi and Shaikha S. Al Muhairi, "Arabic Search Engines Improvement: A New Approach using Search Key Expansion Derived From Arabic Synonyms Structure", Institute of Electrical and Electronics Engineering International Conference, pages: 944 - 951.

[3] Abdusalam F A Nwesri (2005), Seyed M.M. Tahaghoghi and Falk Scholer, "Stemming Arabic Conjunctions and Prepositions", Book: String Processing and Information Retrieval at Springerlink, pages: 206-217.

[4] Al-Shalabi (2005), Riyad; Ghassan Kannan, Iyad Hilat, Ahmad Ababneh, Ahmad Al-Zubi. "Experiments with the Successor Variety Algorithm Using the Cutoff and Entropy Methods", Information Technology Journal, pages: 55-62.

[5] Kazem Taghva (2005), Rania Elkhoury and Jeffrey Coombs, "Arabic Stemming Without A Root Dictionary", International Conference on Information Technology, Institute of Electrical and Electronics Engineering, pages: 152-157.

[6] Larkey, L. S. (2005), Ballesteros, Lisa, and Connell, Margaret E. "Light Stemming for Arabic Information Retrieval", Book: Arabic Computational Morphology- Knowledge-based and Empirical Methods at Springerlink, pages: 221-243.

[7] Mustafa, S. H. (2005), "Character contiguity in N-gram-based word matching: the case for Arabic text searching", Journal of the American Society for Information Science and Technology, pages: 819–827.

[8] Al-Sughaiyer (2004), Imad A. and Ibrahim A. Al-Kharashi. "Arabic Morphological Analysis Techniques: A Comprehensive Survey", Journal of the American Society for Information Science and Technology, pages: 189–213.

[9] Mustafa, S. H. (2004), Al-Radaideh, Q. A. "Using N-grams for Arabic text searching", Journal of the American Society for Information Science and Technology, pages: 1002-1007.

[10] Buckwalter, T. (2003). "QAMUS:Arabic lexicography." http://www.qamus.org/morphology.htm

[11] Savoy, J. (2003), and Rasolofo, Y. "Report on the TREC-11 Experiment: Arabic, Named Page and Topic Distillation Searches", Paper presented at the Eleventh Text REtrieval Conference (TREC 2002).

[12] Mohammad Aljlayl (2002) and S. Beitzel. "IIT at TREC-10", Paper presented at the Tenth Text Retrieval Conference.

[13] Chen, A. (2002), and Gey, F. "Building an Arabic stemmer for information retrieval", In Proceedings of the Eleventh Text REtrieval Conference(TREC 2002), Gaithersburg: National Institute of Science and Technology, pages: 631-639.

[14] Xu, J. (2002), Fraser, A., and Weischedel, R. "Empirical studies in strategies for Arabic retrieval", Conference on Research and development in information retrieval, Association for Computing Machinery, Tampere, Finland, pages: 269-274.

[15] Darwish, K. (2002) and Oard, D.W. "CLIR Experiments at Maryland for TREC-2002: Evidence combination for Arabic-English retrieval", In TREC 2002. Gaithersburg: National Institute of Standards and Technology, pages: 703-710.

[16] Gey, F. C. (2002), and Oard, D. W. "The TREC-2001 Cross-Language Information Retrieval Track: Searching Arabic Using English, French or Arabic Queries", Paper presented at the Tenth Text REtrieval Conference (TREC 2001).

[17] Larkey, L. S. (2002), Ballesteros, Lisa, and Connell, Margaret. "Improving Stemming for Arabic Information Retrieval: Light Stemming and Co-occurrence Analysis", Conference on Research and development in information retrieval, Association for Computing Machinery, Tampere, Finland, pages: 275-282.

[18] Mayfield, J. (2002), McNamee, P., Costello, C., Piatko, C., Banerjee, A., and Hopkins, J. JHU/APL at TREC 2001. "Experiments in Filtering and in Arabic, Video, and Web Retrieval", Paper presented at the Tenth Text REtrieval Conference (TREC 2001).

[19] Mohammed Aljlayl (2002) and Ophir Frieder, "On Arabic Search: Improving the Retrieval Effectiveness via a Light Stemming Approach", The eleventh international conference on Information and knowledge management, Association for Computing Machinery, pages: 340-347.

[20] Imad A. Al-Sughaiyer(2002) and Ibrahim A. Al-Kharashi, "Rule Parser for Arabic Stemmer", The 5th International Conference on Text, Speech and Dialogue, Springer, pages: 11-18,

[21] Conference (TREC 2001).

[22] Chengziang Zhai (2001). "Notes on the Lemur TFIDF model." http://www-2.cs.cmu.edu/~lemur/tfidf.ps

[23] Goweder, A. (2001) and De Roeck, A. "Assessment of a significant Arabic corpus", Arabic. Natural Language Processing Workshop at Association for Computational Linguistics/European Association for Computational Linguistics 2001, Toulouse, France.

[24] Lafferty (2001) and C. Zhai. "Risk minimization and language modeling in information retrieval", In 24th Association for Computing Machinery Special Interest Group on Information Retrieval Conference on Research and Development in Information Retrieval (Special Interest Group on Information Retrieval '01).

[25] Kosinov, S. (2001). "Evaluation of N-grams Conflation Approach in Text-Based Information Retrieval", Eighth Symposium on String Processing and Information Retrieval, Institute of Electrical and Electronics Engineering, pages: 136-142.

[26] Pirkola, A. (2001). "Morphological typology of languages for IR", Journal of Documentation, pages: 330-348.

[27] De Roeck, A. N. (2000) and Al-Fares, W. "A morphologically sensitive clustering algorithm for identifying Arabic roots", Proceedings of the 38th Annual Meeting on Association for Computational Linguistics.

[28] Xu, J. (1998) and Croft, W. B. "Corpus-based stemming using co-occurrence of word variants", Association for Computing Machinery *Transactions on Information Systems*, pages: 61-81.

[29] Al-Kharashi, I. (1994) and Evens, M. W. "Comparing words, stems, and roots as index terms in an Arabic information retrieval system", Journal of the American Society for Information Science and Technology, pages: 548-560.

[30] Robertson, S. (1994). "Okapi at TREC-3", Proceedings of the 3rd Text Retrieval Conference, pages: 109-126.

[31] Popovic, M. (1992) and Willett, P. "The effectiveness of stemming for natural-language access to Slovene textual data", Journal of the American Society for Information Science, pages: 384-390.

[32] Robertson, A. (1992) and Willett, P. "Searching for historical word-forms in a database of 17th-century English text using spelling-correction methods", The 15th Annual International Conference Special Interest Group on Information Retrieval, pages: 256-265.

[33] Harman, D. (1991). "How effective is suffixing?", Journal of the American Society for Information Science, pages 7-15.

[34] Siegel, S. (1988) and Castellan, N. J. "Nonparametric Statistics for the Behavioural Sciences", New York: McGraw-Hill.

[35] Rijsbergen, C. J. (1979) Information Retrieval, Butterworths, London.

[36] Adamson, G. (1974) and Boreham, J. "The use of an association measure based on character structure to identify semantically related words and document titles", Information Storage and Retrieval, pages: 253–260.

Appendices

Appendix A. Corpus Sample
Appendix B. Query Set
Appendix C. Relevance Judgments
Appendix D. Trec_Eval Output
Appendix E. Expanded Queries
Appendix F. Stopword List

Appendix A. Corpus Sample

Sample Document from Arabic Newswire LDC Corpus

<DOC>
<DOCNO>19940520_AFP_ARB.0001</DOCNO>
<HEADER>
ارا٠٠٥١ ٤ ع ٨٥١٠ قبرص /افب-تظخ٢٦ الوكالة الدولية/كوريا الشمالية
</HEADER>
<BODY>
<HEADLINE>
الوكالة الدولية للطاقة الذرية تبلغ مجلس الامن بمخالفات ارتكبتها كوريا &HT;
الشمالية
</HEADLINE>
<TEXT>
<P>
فيينا ٢٠- ٥ (اف ب) - اعلنت الوكالة الدولية للطاقة الذرية فى بيان اصدرته ليل الخميس الجمعة ان كوريا الشمالية بدات تفريغ قلب المفاعل النووى الاختبارى بقوة خمسة ميغاوات فى يونبيون مما يعتبر "انتهاكا لاتفاق الضمانات" (التفتيش) الذى سيرفع الى مجلس الامن.
</P>
<P>
وذكر البيان ان الوكالة استندت الى تقرير فى هذا الخصوص اعده فريق من المفتشين الموجودين حاليا فى كوريا الشمالية وقد اكدوا ان عملية تفريغ قلب المفاعل قد بدات.
</P>
<P>
لكن البيان اشار الى ان عملية التفتيش تبقى ممكنة، واقترح ان يصار فورا الى ايفاد مسوءولين الى كوريا الشمالية لمناقشة الترتيبات الضرورية لاجراءات المراقبة المتعلقة بما يجرى حاليا.
</P>
<P>
يشار الى ان الوكالة الدولية للطاقة الذرية تسعى الى التاكد مما اذا كانت كوريا الشمالية قد عمدت ام لا الى اخفاء الوقود المشع الذى يمكن استخدامه فى انتاج البلوتونيوم الضرورى لصنع السلاح الذرى.
</P>
</TEXT>
<FOOTER>
ستز/ع ق/ موا ٧١ افب ———
</FOOTER>
</BODY>
<TRAILER>
جمت ماي ٤٩ 634002
</TRAILER>
</DOC>

Appendix B. Query Set

Arabic Newswire LDC Query Set

Query No.	Translation	Query
1	Performing Arts and Islamic Institutions in the Arab World	فنون العرض و المؤسسات الاسلامية في العالم العربي
2	Arab consumption of Arab and western Cinema	استهلاك العرب للسينما العربية و الغربية
3	Arts in the Arab World after the Gulf War	الفنون بعد حرب الخليج في العالم العربي
4	Naive painting and Westerners in the Arab world	الرسوم الفطرية و الغربيون في الوطن العربي
5	Traditional craftsmen facing technology	الصناع التقليديون في مواجهة التكنولوجيا
6	Major Arab cities and new ways of Advertising	كبريات المدن العربية و الطرق الحديثة للاشهار
7	Criticism and political poetry in the Arab World	النقد و الشعر السياسي في العالم العربي
8	Arab children and Arts in primary and high schools	الطفل العربي و الفنون في المدارس الابتدائية و الثانوية
9	War victims, plastic surgery and Islam	ضحايا الحرب بين الجراحة التجميلية و الاسلام
10	Polio eradication in the Middle East	القضاء على شلل الأطفال في الشرق الأوسط
11	Measles immunization campaigns in the Middle East	حملات التلقيح ضد الحصبة في الشرق الأوسط
12	Bilharzia/Schistosomaisis prevention in Egypt	منع البلهارسيا في مصر
13	Theater in Egypt	المسرح في مصر
14	Israeli tourism in Jordan	السياحة الاسرائيلية في الأردن
15	Satellite TV in rural areas	القنوات الفضائية في المناطق الريفية
16	Environmental protection laws in Egypt	قوانين حماية البيئة في مصر
17	Israel's nuclear capability	الكفاءة النووية لدى اسرائيل
18	Egyptian-Libyan relations during the 1990s	العلاقات بين مصر و ليبيا أثناء التسعينات
19	Alexandria libraries	مكتبات الاسكندرية
20	Tourism in Cairo	السياحة في القاهرة

Query No.	Translation	Query
21	**Significant archaeological finds in the Dead Sea area**	إكتشافات أثرية هامة في منطقة البحر الميت
22	**Local newspapers and the new press law in Jordan**	الصحف المحلية و قانون الصحافة الجديد في الأردن
23	**Information technology and the Arab World**	عصر المعلومات و العالم العربى
24	**Water resource problems in the Nile Valley**	مشكلات موارد المياه فى حوض وادى النيل
25	**European and American roles in Middle**	الدور الاوروبى والامريكى فى عملية السلام في الشرق الأوسط

Appendix C. Relevance Judgments

A Sample of Relevance Judgments Against LDC Corpus
(Taken from Abu El-Khair, 2003)

Query Number	Document Number		Total Documents
10	19940921_AFP_ARB.0058	19980713_AFP_ARB.0094	57
	19941020_AFP_ARB.0025	19980911_AFP_ARB.0122	
	19941027_AFP_ARB.0147	19990202_AFP_ARB.0001	
	19941219_AFP_ARB.0030	19990316_AFP_ARB.0028	
	19950110_AFP_ARB.0084	19990316_AFP_ARB.0124	
	19950110_AFP_ARB.0085	19990629_AFP_ARB.0158	
	19950125_AFP_ARB.0156	19990630_AFP_ARB.0068	
	19950221_AFP_ARB.0150	19990701_AFP_ARB.0081	
	19950306_AFP_ARB.0095	19990719_AFP_ARB.0126	
	19950307_AFP_ARB.0026	19990920_AFP_ARB.0100	
	19950318_AFP_ARB.0065	19990923_AFP_ARB.0050	
	19950614_AFP_ARB.0062	19990929_AFP_ARB.0092	
	19950614_AFP_ARB.0063	19991003_AFP_ARB.0253	
	19950926_AFP_ARB.0030	19991013_AFP_ARB.0078	
	19951001_AFP_ARB.0089	19991013_AFP_ARB.0105	
	19951001_AFP_ARB.0090	19991029_AFP_ARB.0082	
	19951021_AFP_ARB.0063	20000106_AFP_ARB.0032	
	19960205_AFP_ARB.0090	20000131_AFP_ARB.0104	
	19960304_AFP_ARB.0079	20000313_AFP_ARB.0088	
	19960724_AFP_ARB.0067	20000314_AFP_ARB.0166	
	19970213_AFP_ARB.0068	20000328_AFP_ARB.0068	
	19970311_AFP_ARB.0069	20000421_AFP_ARB.0047	
	19970509_AFP_ARB.0137	20000425_AFP_ARB.0081	
	19970530_AFP_ARB.0096	20000425_AFP_ARB.0108	
	19971015_AFP_ARB.0056	20000516_AFP_ARB.0072	
	19971015_AFP_ARB.0075	20000907_AFP_ARB.0039	
	19971020_AFP_ARB.0090	20001018_AFP_ARB.0022	
	19980312_AFP_ARB.0114	20001022_AFP_ARB.0002	
	19980529_AFP_ARB.0097		

Appendix D. Trec_Eval Output Sample

Treceval Evaluation Stats Output Sample for Run DICE_SIMP_75

Summary Statistics			
Run ID:		DICE_SIMP_75	
Number of Topics:		25	
Total Number of Documents over all Topics			
Retrieved:		25000	
Relevant:		4122	
Rel-Ret:		1633	
Recall Level Precision Averages		**Document Level Averages**	
Recall	**Precision**		**Precision**
0.00	0.7859	At 5 Docs	0.5440
0.10	0.5000	At 10 Docs	0.5200
0.20	0.3980	At 15 Docs	0.4960
0.30	0.3112	At 20 Docs	0.4640
0.40	0.2543	At 30 Docs	0.4187
0.50	0.1513	At 100 Docs	0.2836
0.60	0.1200	At 200 Docs	0.1994
0.70	0.1032	At 500 Docs	0.1112
0.80	0.0300	At 1000 Docs	0.0653
0.90	0.0300		
1.00	0.0300	R-Precision (precision after R (= num_rel for a query) docs retrieved):	
Average precision over all relevant docs			
Non-Interpolated:	0.2175	Exact:	0.2681

Appendix E. Selected Expanded Queries

Expanded Query for Run Id, L10

Qid: 1> فن عرض مؤسس اسلام عالم عرب

Qid: 2> استهلاك عرب سينما عرب غرب

Qid: 3> فن حرب خليج عالم عرب

Qid: 4> رسوم فطر غرب وطن عرب

Qid: 5> صناع تقليد مواجه تكنولوجيا

Qid: 6> كبر مدن عرب طرق حديث اشهار

Qid: 7> نقد شعر سياس عالم عرب

Qid: 8> طفل عرب فن مدارس ابتداء ثانو

Qid: 9> ضحايا حرب جراح تجميل اسلام

Qid: 10> قضاء شلل اطفال شرق اوسط

Qid: 11> حمل تلقيح حصب شرق اوسط

Qid: 12> منع بلهارسيا مصر

Qid: 13> مسرح مصر

Qid: 14> سياح اسرائيل اردن

Qid: 15> قنو فضاء مناطق ريف

Qid: 16> قوان حما بيء مصر

Qid: 17> كفاء نوو اسرائيل

Qid: 18> علاق مصر ليبيا اثناء تسع

Qid: 19> مكتب اسكندر

Qid: 20> سياح قاهر

اكتشاف اثر هام منطق بحر ميت <Qid: 21

صحف محل قان صحاف جديد اردن <Qid: 22

عصر معلوم عالم عرب <Qid: 23

مشكل موارد ميا حوض اد نيل <Qid: 24

دور اوروب امريك عمل سلام شرق اوسط <Qid: 25

Expanded Query for Run Id, NoST_NG

Qid: 1

فنون وفنون لفنون فنوني بفنون العرض بالعرض والعرض العرضي لعرض المؤسسات والمؤسسات المؤسساتي بالمؤسسات فالمؤسسات لمؤسسات المؤسات المؤسساتيه للمؤسسات الاسلاميه والاسلاميه الاسلامي اسلاميه لاسلاميه الاستسلاميه الاسلامين بالاسلامي والاسلامي واسلاميه العالم لعالم للعالم العالمه والعالم العالمي فالعالم بالعالم عالم العا العال العربي العربيه العربيد والعربي بالعربي العرب

Qid: 2

لاستهلاك استهلاك الاستهلاك باستهلاك للاستهلاك استهلاكه واستهلاك استهلاكا الاستهلاكي استهلاكها استهلاكيه استهلاكهم والاستهلاك لاستهلاكها العرب العربي بالعرب فالعرب العربه والعرب لعرب للسينما وللسينما لسينما للسينمائي العربيه االعربيه والعربيه بالعربيه العربييه العربي لعربيه الغربيه والغربيه الغربي لغربيه

Qid: 3

الفنون بالفنون والفنون لفنون حرب الخليج والخليج الخليجي بالخليج لخليج العالم لعالم للعالم العالمه والعالم العالمي فالعالم بالعالم عالم العا العال العربي العربيه العربيد والعربي بالعربي العرب

Qid: 4

الرسوم والرسوم بالرسوم الرسو لرسوم الفطريه الفطري الغربيون والغربيون الوطن الوطني بالوطن والوطن العربي العربيه العربيد والعربي بالعربي العرب

Qid: 5

الصناع الصناعه الصناعي التقليديون التقليدي تقليديون مواجهه لمواجهه بمواجهه ومواجهه واجهه مواجه التكنولوجيا والتكنولوجيا التكنولوجي بالتكنولوجيا التكنولوجيات لتكنولوجيا والتكنولوجيات والتكنولوجي التكنولوجيه للتكنولوجيا تكنولوجيا للتكنولوجيات والتكنولوجيه

Qid: 6

كبريات كبرياتي المدن المدني والمدن بالمدن المد لمدن العربيه االعربيه والعربيه بالعربيه العربييه العربي لعربيه الطرق والطرق بالطرق الطرقي لطرق الحديثه والحديثه الحديث

Qid: 7

النقد النقدي بالنقد والنقد الشعر والشعر الشعري بالشعر فالشعر الشعره السياسي السياس السياسيه السياسين السياسيان السياسيات السياسه والسياسي العالم لعالم للعالم العالمه والعالم العالمي فالعالم بالعالم عالم العا العال العربي العربيه العربيد والعربي بالعربي العرب

Qid: 8

الطفل والطفل الطفله بالطفل لطفل الطف العربي العربيه العربيد والعربي بالعربي العرب الفنون بالفنون والفنون لفنون المدارس والمدارس بالمدارس المدار لمدارس الابتدائيه الابتدائي ابتدائيه الثانويه والثانويه الثانوي والثانوي

Qid: 9

ضحايا لضحايا ضحاياه وضحايا بضحايا الحرب بالحرب والحرب الحربه الحربي فالحرب الحربش الحر لحرب الجراحه والجراحه الجراح لجراحه التجميليه التجميل تجميليه الاسلام الاسلامي فالاسلام بالاسلام والاسلام لاسلام اسلام

Qid: 10

القضاء والقضاء فالقضاء بالقضاء لقضاء شلل الاطفال اطفالا لاطفال والاطفال الاطفاء للاطفال بالاطفال واطفالا لاطفاله كالاطفال اطفال الشرق بالشرق الشرقي والشرق الشر لشرق الاوسط والاوسط الاووسط لاوسط

Qid: 11

حملات حملاته لحملات لاتحمل وحملات بحملات حملا ملات التلقيح والتلقيح بالتلقيح لتلقيح التلقي الحصبه بالحصبه والحصبه الشرق بالشرق الشرقي والشرق الشر لشرق الاوسط والاوسط الاووسط لاوسط

Qid: 12

منع البلهارسيا بالبلهارسيا للبلهارسيا بلهارسيا مصر

Qid: 13

المسرح المسرحي والمسرح بالمسرح لمسرح مصر

Qid: 14

السياحه والسياحه بالسياحه السياح الاسرائيليه الاسرائيلي الاسرائيبيه والاسرائيليه لاسرائيليه لالاسرائيلي والاسرائيلي الاسرائيليي الاسرائيلين الاسرائيل اسرائيليه لاسرائيلي الاسرائيليان والاسرائيلين الاسرائيلين والاسرائيليي الاسرائيليين الاسرائيليين الاسرائيليات الاسرائيليون الاردن والاردن الاردني كالاردن فالاردن االاردن بالاردن لاردن

Qid: 15

القنوات بالقنوات الفضائيه والفضائيه الفضائي والفضائي المناطق بالمناطق والمناطق فالمناطق لمناطق المناط،الريفيه والريفيه الريفي

Qid: 16

قوانين وقوانين قوانينه بقوانين لقوانين وانين حمايه وحمايه بحمايه لحمايه البيئه والبيئه بالبيئه مصر

Qid: 17

الكفاءه والكفاءه بالكفاءه لكفاءه النوويه النوويه والنوويه النووي اسرائيل اسرائيلي واسرائيل لاسرائيل فاسرائيل باسرائيل اسرائيل اسرائيله كاسرائيل سرائيل اسرائي اسرائيليا اسرائيليه ولاسرائيل باسرائيلي اسرائيلين اسرائيلوف لاسرائيلي االسرائيلي الاسرائيل واسرائيلي اسرائيليي

Qid: 18

العلاقات والعلاقات بالعلاقات كالعلاقات فالعلاقات لعلاقات وبالعلاقات للعلاقات ولعلاقات لعلاقاته مصر ليبيا وليبيا ليبيا كليبيا لليبيا بليبيا ليبيان ليبي اثناء واثناء فاثناء اثناءه اثنا التسعينات والتسعينات التسعين

Qid: 19

مكتبات ومكتبات مكتبا الاسكندريه والاسكندريه بالاسكندريه الاسكندري للاسكندريه الاسكندر اسكندريه

Qid: 20

السياحه والسياحه بالسياحه السياح القاهره بالقاهره والقاهره فالقاهره لقاهره القاهر القاهرهان

Qid: 21

اكتشافات واكتشافات لاكتشافات اكتشافاته اكتشافا الاكتشافات اكتشافاتها اكتشافاتهم اثريه واثريه اثري ثريه هامه مهامها مهامه مهام امها ومهام مهامهم امامها بمهام وامها بمهامها بمهامه عامها مهامي امهال وهامه مهاما لمهامها امهات ومهامها لمهامه تهامه سهامها مهامك منطقه بمنطقه منطقها ومنطقه لمنطقه منطقهم كمنطقه منطق البحر البحري بالبحر والبحر فالبحر لبحر البح الميت الميته المميت والميت الميتمي المي لميت

Qid: 22

الصحف والصحف الصحفي بالصحف فالصحف لصحف المحليه والمحليه المحلي قانون فقانون وقانون قانونا قانوني بقانون لقانون الصحافه والصحافه بالصحافه الصحاف الجديد الجديدي الجديده الجديدين والجديد الجدي الاردن والاردن الاردني كالاردن فالاردن االاردن بالاردن لاردن

Qid: 23

عصر المعلومات والمعلومات المعلوماتي بالمعلومات لمعلومات المعلوماتيه والمعلوماتي للمعلومات معلومات ومعلومات المعلوم المعلمات العالم لعالم للعالم العالمه والعالم العالمي فالعالم بالعالم عالم العا العال العربي العربيه العربيد والعربي بالعربي العرب

Qid: 24

مشكلات لمشكلات مشكلاته بمشكلات ومشكلات مشكلا موارد موارده بموارد وموارد لموارد وارد موار المياه بالمياه والمياه كالمياه لمياه حوض وادي جوادي واديب ووادي واديس وادين لوادي نوادي بوادي توادي النيل بالنيل والنيل لنيل نيلن النيي

Qid: 25

الدور الدوري الدوره بالدور والدور فالدور لدور الدو الاوروبي والاوروبي الاوروبيه الاوروبين لاوروبي الاوروبيين والاوروبيه الاوروبيون الاوروبيات الاوروربي الاوروربي لاوروبيه الامريكي الامريكيه عمليه لعمليه كعمليه وعمليه بعمليه فعمليه عمليه اعمليه عمليها عملي السلام والسلام السلامه فالسلام بالسلام السلامي السلال لسلام الشرق بالشرق الشرقي والشرق الشر لشرق الاوسط والاوسط الاووسط لاوسط

Expanded Query for Run Id, L10_NG / NG_DIC_SMP /NG_DIC_SMP_085

Qid: 1
فن عرض مؤسس مؤسست لمؤسس بمؤسس مؤسسا كمؤسس مؤسسو اسلام لاسلام باسلام كاسلام اسلامي اسلاما سلام سلاس سلاسل عالم معالم عالما لعالم لمعال بعالم كعالم عرب

Qid: 2
استهلاك لاستهلاك استهلاكا باستهلاك استهلا استهلاكهم عرب سينما سينمائ لسينما بسينما سينم عرب غرب

Qid: 3
فن حرب خليج بخليج لخليج عالم معالم عالما لعالم لمعال بعالم كعالم عرب

Qid: 4
رسوم مرسوم رسوما برسوم لرسوم كرسوم سومرس فطر غرب وطن عرب

Qid: 5
صناع صناعي لصناع صناعت بصناع تقليد بتقليد تقليدا لتقليد مواجه لمواجه بمواجه مواجهت مواج واجه تكنولوجيا لتكنولوجيا بتكنولوجيا كنولوجيا تكنولوجيز تكنولوجيت تكنولوج

Qid: 6
كبر مدن عرب طرق حديث تحديث حديثا بحديث حديثو لحديث اشهار باشهار

Qid: 7
نقد شعر سياس سياسيا اسياس اسيا سياسي اسياد سياست اساسيا لسياس بسياس قياسيا لاسيا قاسيا سياسف سياسيت ماسياس سسياس كسياس راسيا باسيا داسيا لاسياس ياسيف كاسيا كاسياس عالم معالم عالما لعالم لمعال بعالم كعالم عرب

Qid: 8
طفل عرب فن مدارس لمدارس بمدارس مدار دارس ابتدائ ابتدا ثانو

Qid: 9
ضحايا لضحايا بضحايا حرب جراح بجراح لجراح جراحا جراحت تجميل لتجميل جميل اسلام لاسلام باسلام كاسلام اسلامي اسلاما سلام سلاس سلاسل

Qid: 10
قضاء لقضاء بقضاء شلل اطفال اطفالا لاطفال باطفال اطفا شرق اوسط لاوسط

Qid: 11
حمل تلقيح لتلقيح بتلقيح حصب شرق اوسط لاوسط

Qid: 12
منع بلهارسيا مصر

Qid: 13
مسرح مسرحا لمسرح بمسرح كمسرح مصر

Qid: 14

سياح سياحيا سياحا لسياح بسياح سياحت اسرائيل لاسرائيل اسرائيلب اسرائيلي اسرائيلت باسرائيل االسرائيل اسرائييل كاسرائيل فاسرائيل سرائيل اسراني اسرائيليا اسرائيليت لالاسرائيل اسرائيلوف اردن اردنا غاردن لاردن اردني باردن هاردن

Qid: 15

قنو فضاىئ فضائح فضائل مناطق بمناطق لمناطق مناطقنا ناطق مناط ريف

Qid: 16

قوان لقوان قوانا بقوان حما بيئ مصر

Qid: 17

كفاء بكفاء كفاءت لكفاء اكفاء نوو نووو اسرائيل لاسرائيل اسرائيلب اسرائيلي اسرائيلت باسرائيل االسرائيل اسرائييل كاسرائيل فاسرائيل سرائيل اسراني اسرائيليا اسرائيليت لالاسرائيل اسرائيلوف

Qid: 18

علاق علاقت لاعلاق بعلاق لعلاق اعلاق مصر ليبيا ليببيا بليبيا فليبيا كليبيا يبيا ليبي ليبيب اثناء فاثناء اثنا ثناء تسع

Qid: 19

مكتب مكتبا لمكتب كمكتب بمكتب مكتبت مكتتب اسكندر سكندر

Qid: 20

سياح سياحيا سياحا لسياح بسياح سياحت قاهر لقاهر

Qid: 21

اكتشاف لاكتشاف باكتشاف اكتشافا اثر هام منطق منطقت بمنطق لمنطق كمنطق بحر ميت

Qid: 22

صحف محل قان صحاف صحافت كصحاف لصحاف صحافي بصحاف جديد بجديد جديدا تجديد جديدت لجديد اردن اردنا غاردن لاردن اردني باردن هاردن

Qid: 23

عصر معلوم بمعلوم معلوما لمعلوم علوم عالم معالم عالما لعالم لمعال بعالم كعالم عرب

Qid: 24

مشكل لمشكل مشكلت كمشكل بمشكل مشكلا موارد لموارد بموارد وارد موار ميا حوض اد نيل

Qid: 25

دور اوروب اوروبا اورروب باوروبا لاوروب اوروبب اوروبي اورورو اورو امريك امريكا امريكت عمل سلام اسلام بسلام سلامت سلاما سلامو لاسلام لسلام شرق اوسط لاوسط

Expanded Query for Run Id, NG_DIC_DIF /NG_DIC_DIF_060

Qid: 1

فن عرض تعرض يعرض عرضت معرض عرضا اعرض مؤسس مؤس مؤسست مؤسسا لمؤسس بمؤسس كمؤسس سالام اسلام باسلام كاسلام سالما اسلاما لاسلام مالام عالم الم معالم اعلم عرب اعرب تعرب عربت يعرب عربا معرب

Qid: 2

استهلاك باستهلاك استهلاكا لاستهلاك استهلكت استهلك عرب اعرب تعرب عربت يعرب عربا معرب سينام سينما سينمائ سيام عرب اعرب تعرب عربت يعرب عربا معرب غرب مغرب اغرب غربا غربت غربب

Qid: 3

فن حرب حربا حربت خليج لخليج عالم الم معالم اعلم عرب اعرب تعرب عربت يعرب عربا معرب

Qid: 4

رسوم رسوما رسو روسومو فطر افطر يفطر فطرا غرب مغرب اغرب غربا غربت غربب وطن عرب اعرب تعرب عربت يعرب عربا معرب

Qid: 5

صناع صناعت صنا صنعا تقيليد تقاليد تقليد مقاليد تقليدا بتقاليد لتقاليد بتقليد لتقليد تيليد توليد بتوليد لتوليد قلايد مواجه واجه تواجه يواجه لمواجه بمواجه مواجهت تكنولوجيا كتنولوجيا لتكنولوجيا بتكنولوجيا تكنولوجيت كنولوجيا اتنولوجيا تكنولوجيز

Qid: 6

كبر اكبر يكبر مكبر تكبر كبرت مدن تمدن امدن عرب اعرب تعرب عربت يعرب عربا معرب طرق تطرق طرقا مطرق يطرق طرقت حديث احاديث حديثا شاهار اشهار باشهار شاهرا شاهبهار اشهرا

Qid: 7

نقد نقدا شعر يشعر شعرت تشعر اشعر شعرا مشعر يسا سياس اساياس سياست اسياس سياسي ماسياس عالم الم معالم اعلم عرب اعرب تعرب عربت يعرب عربا معرب

Qid: 8

طفل طفلا طفلت تطفل عرب اعرب تعرب عربت يعرب عربا معرب فن مدارس دارس تدارس لمدارس بمدارس ابتدائ باداائ بارتدائ ثانو انو

Qid: 9

ضحايا ضحا لاضحايا حرب حربا حربت جراح جرا جراحا جراحت جرحا جيميل تجميل جاميل جيميلل تاميل لتجميل ستيميل تيميل جاميلل سالام اسلام باسلام كاسلام سالما اسلاما لاسلام مالام

Qid: 10

قضاء قضا شلل شللا شلال لشلل شلول شلالا اطفال لاطفال باطفال اطفالا شرق شرقا مشرق تشرق اواسط واسط وسط اوسط اوساط لاوسط اوساطا اوس اسط وسطا واطس

Qid: 11

حمل تحمل يحمل محمل حملت احمل حملا تلقيح لتلقيح تلويح بتلقيح حصب شرق شرقا مشرق تشرق اواسط واسط وسط اوسط اوساط لاوسط اوساطا اوس اسط وسطا واطس

Qid: 12
منع نمنع يمنع تمنع امنع بلهارسيا مصر

Qid: 13
مسرح لمسرح بمسرح كمسرح مسرحا يسرح سرح مسر تسرح مصر

Qid: 14
سياح سياحا سياحت االسرائيل اسرائيل فاسرائيل لاسرائيل كاسرائيل باسرائيل لالاسرائيل اسرائيلي اسرائيلت سرائيل اسرائييل يسرائيل اسرائلي اسيرائيل اسرنايل اردن ارد اردنا ردن يردن

Qid: 15
قنو فضائ فضا مناطق ناطق بمناطق لمناطق رايف ريف ماريف يوريف ياريف يرف يف ري يريف ريفا شيريف ديريف تريف ريفت

Qid: 16
قوان قوانا حما بيئ بي بينت مصر

Qid: 17
كفاء كفا كفاءت لاكفاء نوو نو انوو وو نووو االسرائيل اسرائيل فاسرائيل لاسرائيل كاسرائيل باسرائيل لالاسرائيل اسرائيلي اسرائيلت سرائيل اسرائييل يسرائيل اسرائلي اسيرائيل اسرنايل

Qid: 18
علاق علاقت علا لاعلاق لعلاق علقا مصر ليايا ليبيا ليببيا اثناء فاثناء تسع ستسع ستتسع يتسع متسع اتسع

Qid: 19
مكتب كمكتب لمكتب بمكتب كتب يكتب مكتبا تكتب اكتب مكتبت مكتتب مكت اسكندر سكندر

Qid: 20
سياح سياحا سياحت قاهر اهر

Qid: 21
اكتشاف لاكتشاف باكتشاف اكتشافا اكتشافات اكتشفا اثر ماهام هام اهم هما ام ها اهما هاما مهام اهام تهام باهام هامت منطق كمنطق بمنطق لمنطق منطقت نطق ينطق تنطق انطق منط بحر بحرا تبحر ابحر ببحر مبحر يبحر ميت يمت مي يت

Qid: 22
صحف اصحف صحفا مصحف محل قان قنا اقن قا ان قانا مقان خاقان صحاف صحافت صحا صحفا جديد جديدا جديدت جدد جددت جددا اردن ارد اردنا ردن يردن

Qid: 23
عصر عصرا معلوم معلومات يعالوم معلمو لمعلوم معلوما بمعلوم بمعلومات عالم الم معالم اعلم عرب اعرب تعرب عربت يعرب عربا معرب

Qid: 24

مشكل لمشكل كمشكل بمشكل يشكل تشكل شكل مشكلت مشكلا اشكل موارد وارد لموارد بموارد هاوارد ميا يما يا مي ماي امايا حوض وحض حو حوضا اد نويل اينيل نيل مينيل نيويل نيال نايل ايونيل مانيل يونيل نيالا منايل اونيل نينيل ينل يل ني نيلت كينيل دينيل انيل فينيل زينيل منيل نيلل نيوفيل ليونيل نيوويل

Qid: 25

ودرو دور دوراو اودور دوار تودور دوارت دورات مداور مادور دورو دورادو ودر درو دو ور مدور هودور ادور دورا يدور دورت تدور مودرو دورر دوراسو اوروب لاوروب اورروب يوروب اوروبا اوروبب اوربو اميريك امريك اميركي ماريك اميريكا امريكت امريكا ميريك سميريك ايريك اوريك عمل تعمل يعمل اعمل معمل عملا عملت سالام سلام سلا سلاما سلامت لاسلام كاسلام باسلام لسلام سلما شرق شرقا مشرق تشرق اواسط واسط وسط اوسط اوساط لاوسط اوساطا اوس اسط وسطا واطس

Expanded Query for Run Id, NG_JAC_BND /NG_JAC_BND_060

Qid: 1

فن ففن فنن فنحن فنان فنون فنسن عرض مؤسس مؤس اسلام اسلامهم استسلام لاسلام كاسلام استلام اسلامي اسلاما باسلام استسلامهم عالم عرب

Qid: 2

استهلاك لاستهلاك استهلاكا باستهلاك استهلك استهلا استهلاكهم استملاك استهلال عرب سينما سينمائيا سينمائ لسينما بسينما عرب غرب غربب غربيب

Qid: 3

فن ففن فنن فنحن فنان فنون فنسن حرب خليج عالم عرب

Qid: 4

رسوم فطر غرب غربب غربيب وطن عرب

Qid: 5

صناع تقليد بتقليد تقاليد تقليدا لتقليد تقيليد مواجه متواجه لمواجه بمواجه مواجهت تكنولوجيا لتكنولوجيا بتكنولوجيا كنولوجيا تكنلوجيا تكنولوجيز تكنولوجيت تكنولوج

Qid: 6

كبر كاكبر مدن متمدن عرب طرق حديث اشهار باشهار

Qid: 7

نقد شعر سياس سسياس سياسي سياسيا اسياس عالم عرب

Qid: 8

طفل عرب فن ففن فنن فنحن فنان فنون فنسن مدارس لمدارس بمدارس ابتدائ ابتدا ابدائ ثانو

Qid: 9
ضحايا ضحا ضحاياهما لضحايا بضحايا حرب جراح تجميل لتجميل اسلام اسلامهم استسلام لاسلام كاسلام استلام اسلامي اسلاما باسلام استسلامهم

Qid: 10
قضاء شلل شل ششل اطفال باطفال اطفالا لاطفال شرق اوسط

Qid: 11
حمل تلقيح لتلقيح بتلقيح حصب شرق اوسط

Qid: 12
منع بلهارسيا مصر

Qid: 13
مسرح مصر

Qid: 14
سياح اسرائيل اسرائييل االسرائيل لاسرائيل اسرائيلي اسرائيلت اسرائيسل باسرائيل اسرائيلب اسيرائيل كاسرائيل فاسرائيل اسرائييل اسرايل اسرائي اسرئيل سرائيل اسرائل ارائيل اسرائيليت اسرائيليا لالاسرائيل اسرائيلوف اشرائيل اسرائبل يسرائيل اردن االاردن

Qid: 15
قنو فضائ مناطق مناطقنا بمناطق لمناطق ريف ريسيف ريغيف ريليف ريشيف

Qid: 16
قوان قوانين حما حماما حماسا حميما حمارا بيئ مصر

Qid: 17
كفاء نوو نووو نو نونو ننو اسرائيل اسرائييل االسرائيل لاسرائيل اسرائيلي اسرائيلت اسرائيسل باسرائيل اسرائيلب اسيرائيل كاسرائيل فاسرائيل اسرائييل اسرايل اسرائي اسرئيل سرائيل اسرائل ارائيل اسرائيليت اسرائيليا لالاسرائيل اسرائيلوف اشرائيل اسرائبل يسرائيل

Qid: 18
علاق مصر ليبيا ليببيا ليبيريا ليبيبا ليا ليبريا فليبيا بليبيا ليبايا ليبيرا كليبيا اثناء فاثناء اسثناء تسع تتسع

Qid: 19
مكتب مكتتب اسكندر سكندر اسكندران

Qid: 20
سياح قاهر

Qid: 21
اكتشاف لاكتشاف باكتشاف اكتشافا اكتشف اكتاف اكتشافهم اكتشافات اثر اتاثر هام هايام منطق بحر ببحر بحرار ميت مميت ميريت محميت ماميت ميكيت

Qid: 22

صحف محل محلل محلول قان قانون صحاف جديد جد اردن االاردن

Qid: 23

عصر معلوم لمعلوم بمعلوم معلوما معلوماتهم عالم عرب

Qid: 24

مشكل موارد بموارد لموارد ميا ميارا مسميا ميالا مغميا ميديا محميا مينيا ميريا ميخيا ميليا حوض اد ادد اياد اكاد افاد اجاد ازاد اعاد اماد احاد اشاد اباد اراد اناد اديد اواد نيل نينيل نيلل نييل نيفيل نيغيل نيويل

Qid: 25

دور دورر دردور دورير دولور دوفور دومور دورنر دوهور اوروب اورروب اوروبب اوروروب اوروورب اوب اوروبا لاوروب اورويب اوروبي امريك امريكا اميريك امريكت عمل سلام سلامهم سلامتهم سلامتكم شرق اوسط

Appendix F. Stopword List

From Light10_Stop Function of Lemur Toolkit

"ان","بعد", "ضد", "يلي", "الى", "في", "من", "حتى", "وهو", "يكون"
به", "وليس", "أحد", "على", "وكان", "تلك", "كذلك", "التي",
فيها", "عليها", "إن", "وعلى", "لكن", "عن", "مساء", "ليس", "","وبين
الذي", "أما", "حين", "ومن", "لا", "ليسب", "وكانت", "أي", "","منذ
حول", "دون", "مع", "لكنه", "ولكن", "له", "هذا", "","ما", "عنه
هذه", "أنه", "تكون", "قد", "بين", "جدا", "لن", "","والتي","فقط", "ثم
لأن", "اليوم", "لم", "هؤلاء", "فإن", "فيه", "ذلك", "","نحو", "كان", "لهم
اللذين", "كل", "بد", "لدى", "وثي", "أن", "ومع", "","لو", "عند
عنها", "منه", "بها", "وفي", "فهو", "تحت", "لها", "","فقد", "بل", "هو
عليه", "كما", "كيف", "هنا", "وقد", "كانت", "","أو", "إذ", "علي
قبل", "معه", "يوم", "منها", "إلى", "إذا", "","لذلك", "أمام", "هناك
او", "و", "ما", "لا", "الي", "إلي", "","هل", "حيث", "هي", "اذا
مايزال", "اصبح", "أصبح", "أمسى", "امسى", "","مازال", "لازال", "لايزال
مابرح", "مافتئ", "ما انفك", "بات", "صار", "","أضحى", "اضحى", "ظل
ليت", "لعل", "لاسيما", "ولايزال", "الحالي", "ضمن", "","ليس", "إن", "كأن
ذات", "اي", "بدلا", "اليها", "انه", "الذين", "فانه", "","اول", "وله
","والذي", "وهذا", "لهذا", "الا", "فكان", "ستكون", "مما", "أبو","وان
بإن", "الذي", "اليه", "يمكن", "بهذا", "لدي", "وأن", "وهي", "
"آل", "الذي", "هن", "الذى","وأبو