Enhancing Retrieval Effectiveness of Diacritisized Arabic Passages Using Stemmer and Thesaurus

Bassam Hammo Azzam Sleit Mahmoud El-Haj

Jordan University
King Abdullah II School for Information Technology
Computer Information Systems
Amman, 11942 Jordan

{b.hammo, azzam.sleit}@ju.edu.jo, madhajj@hotmail.com

Abstract

In this paper we discuss the enhancement of Arabic passage retrieval for both diacritisized and non-diacritisized text. Most previous work suggested that retrieval start with pre-processing the Arabic text to remove the diacritical marks (short vowels) to unify the text. In most cases, this process causes considerable ambiguity at the word level in the absence of context. However, searching for a word in diacritisized text requires typing and matching all its diacritical marks, which is cumbersome and prevents users from searching and hence retrieving valuable amount of text. The other way around, is to ignore these marks and fall into the problem of ambiguity. In this paper, we propose a passage retrieval approach to search for diacritic and diacritic-less text through query expansion to match a user's query. We applied a rule-based stemmer and we compiled a huge thesaurus for this purpose. We tested our approach on the scripts of the Quran as an open domain source of diacritisized text using a set of 40 non-diacritical words obtained from testers. The results are presented and the applied approach reveals future directions for search engines.

Introduction

Arabic is a Semitic language. Like Hebrew, the orthography of the Arabic language has two main parts: characters that represent the consonant sounds, and diacritical marks that represent the short vowels and cause variations in pronunciation. The diacritics are written above and below the characters. For instance according to (Kirchhoff and Vergyri 2005), the root كتب (ktb) "to write" with the presence of the diacritics (short vowels) has 21 valid interpretations. For example, it could be interpreted using the pattern (made of the three consonant root and short vowels), فَعَلَ (fa3ala) (pronounced كَتَبَ "he wrote"), or using the pattern فعُلٌ(fo3oln) (pronounced كتُبٌ "books"). An analysis of 23,000 Arabic scripts by (Debili et al. 2002) showed that there is an average of 11.6 possible ways to assign diacritics for every non-vowelized word. However, modern Arabic scripts that appear on the Internet and most of the books aiming grownups are written without diacritics, which causes ambiguity even for educated readers. The only way to disambiguate the words is to locate the words within the context. Diacritics are used, however, in the Quran, religious scripts, books for children, children with reading difficulties, and books for foreign students learning Arabic. The major diacritic-less Arabic scripts available on the Internet will prevent two groups of people from accessing their contents. The first group is visually impaired people, while the second is people with learning disabilities. Both groups rely on computer-based applications of text synthesis and voice recognition. Unfortunately, the success of the Arabic voice applications is highly dependent on the presence of vowelized text which enables the systems to pronounce the words correctly without any ambiguity. Many research projects have been carried out to restore diacritics automatically to help such applications (El-Sadany and Hashish 1988; Gal 2002; Emam and Fisher 2004; Zitouni et al. 2006). In addition, research has been also done to improve Arabic Information Retrieval (IR) through deploying techniques and methodologies as computational morphology among others to improve the recall and the precision (Hmeidi et al. 1997; Abu-Salem et al. 1999; Alsamara et al. 2003; Zitouni et al. 2005). Unfortunately, most researches in the field of Arabic IR did not pay much attention to the problem of searching and retrieving vowelized text. Most published works even suggested removing the diacritics in the preprocessing step and unifying the content of the inverted list (Buckwalter 2002). This is because the majority of the text available on the web is not-vowelized (or using few diacritical marks). In addition, typing and matching words with diacritics is cumbersome.

The Search Engine

An Information Retrieval Engine to experiment with Arabic document retrieval is described in (Hammo et al. 2002). Later, this engine has been modified to support passage retrieval for an Arabic open-domain question answering system (Hammo et al. 2004).

Copyright © 2008, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

Like most of the current search engines, our system is based on the famous vector space model (Salton 1983; Salton 1989). It takes a query in the Arabic language and attempts to provide a ranked list of passages as an output. The main components of the system include: *Splitter*, *Tokenizer*, *Stemmer* and *Thesaurus*. The data flow of our system is depicted in Figure 1.

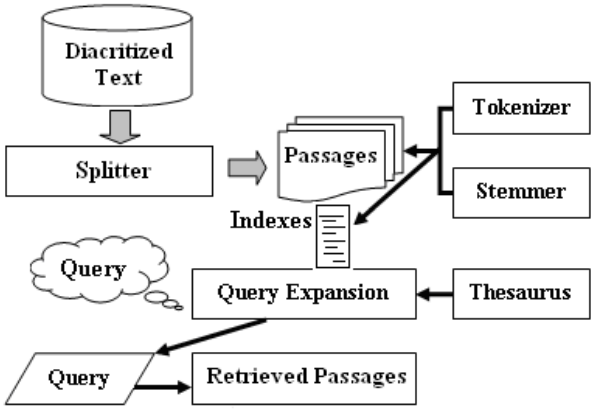

Figure 1. Processing Modules and Data Flow

The Domain of the Study

To test our approach for retrieving diacritisized Arabic text, we have chosen the holy Quran. The scripts of the Quran are vowelized to preserve the spelling and the meaning of its words, which tremendously can be changed by the vowels. Our approach is simply based on searching the words of the Quran without being worried about typing the diacritics. We improved the search process through automatic query expansion using a stemmer and a thesaurus. Users need only to type in the words he want to search for and their queries are automatically augmented with all morphological variants of the query's words to expand the search. Also, we investigated the effectiveness of applying a thesaurus of semantic classes to expand the search. The results obtained are promising and open directions for enhancing the capabilities of search engines, and other applications, such as, question answering, and information extraction from the Quran.

Data Acquisition

First, the Quran chapters (*suras*) are divided into verses (*ayat*) using the *Splitter* module. The *Tokenizer* divides the verses into words (tokens), while a rule-based *Stemmer* uses morphological heuristics to determine the morphological root of a given word. In this system we provide three types of indexes: the Vowelized-Word Index, the Non-Vowelized-Word Index, and the Root Index, which are used to manipulate the data. Finally, a *Thesaurus* of semantic word classes is used to expand the user's query. In the following sections we describe the implementation and the basic processing outline of the search engine.

The Quran consists of 114 chapters (suras) and contains a total of 6236 verses (ayat). Each surah is generally known by a name. Table 1 shows some statistics about the Holy Quran.

Table 1. Statistics of the Holy Quran

Number of pages	604
Number of chapters	114
Number of parts	30
Number of sections	60
Number of verses	6236
Number of words	77845
Number of distinct vowelized words	19273
Number of distinct non-vowelized words	14918
Number of distinct roots*	1767
* also includes proper names of nonArabic origins	

Implementation of the Search Engine

An Information Retrieval (IR) system can be constructed using a relational database management system (RDBMS) (Lundquist et al. 1997). Designing an IR system on the relational model retains the benefits of providing fast and sophisticated retrieval, being portable across platforms as well as being able to use the security and integrity features built in the RDBMS system (Lundquist et al. 1997). In our work, we adapted the idea of integrating an RDBMS with an IR system to store and manipulate the Quran scripts. The data model of the engine is depicted in Figure 2. It contains the following database relations:

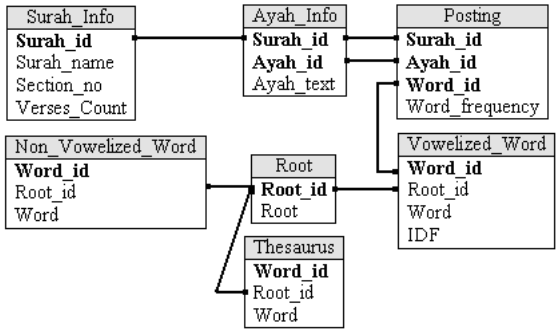

Figure 2. The Relational Database Model

- Surah_Info_Table(Surah_id, Surah_name, Section_no, Verses_count): to store chapter information (one row per surah (chapter)).
- Ayah_Info_Table (Surah_id, Ayah_id, Ayah_text): to store all the verses extracted from the Quran collection (one row per verse).
- Vowelized_Word_Table (Word_id, Root_id, Word, IDF): to store all distinct vowelized words from the Quran collection, the Root_id and the inverse document frequency (IDF) of each word (one row per word).

- Non_Vowelized_Word_Table (Word_id, Root_id, Word): to store all distinct non-vowelized words from the Quran collection, the Root_id of each word (one row per word).
- Root_Table (Root_id, Root): to store the distinct roots of the words from the Quran collection (one row per root).
- Posting_Table (Surah_id, Ayah_id, Word_id, Word_frequency): to post all occurrences of the words extracted from the Quran scripts (one row per posted word).
- Thesaurus_Table (Word_id, Root_id, Word): to store semantic classes of words from the Quran scripts (one row per word).

Processing the Text of the Quran

Preprocessing the Scripts

The scripts of the Quran, in its current format, go back to the *Rashedi Caliph Othman Bn Affan*, who ordered the companions of the Prophet Mohammad to gather the written scripts in one book. This is the book that all Muslims are reading today with its original format as it was revealed. To make the Quran suitable for searching, preprocessing the verses was absolutely essential before indexing. First, we start with obtaining the verses from the *Splitter* and getting rid of the verse number and the ۝ (U+06DD) symbol, which marks the end of the verse. Other symbols like (۞ ۩) are used to organize the Quran into parts and sections and thus have to be removed. Finally, a set of letters and marks such as (ۜ ۚ ۙ ۘ ۗ ۖ) which are used for reciting the Quran must also be removed before the tokenization process takes place.

Tokenization

The tokenization process starts with acquiring each surah (chapter) from the Surah_Info_Table. Next the *Splitter* module is triggered to split each chapter into verses at the verse boundary symbol "۝". The extracted verses are then tokenized to extract their words. We designed and implemented a tokenizer that extracts words at any white space or punctuation characters. Since the Quran words are bounded only by spaces, the tokens were extracted easily and correctly.

Building the Inverted Lists

The indexes of the search engine are built during the tokenization process. The system builds three indexes as tokens are extracted: one index for the vowelized words, a second one for the non-vowelized words and a third one for roots.

The Vowelized-Words Index

This inverted list contains all distinct words (along with their frequencies), which were obtained from the *tokenizer* module and inserted in this index without preprocessing (i.e., keeping their diacritics intact). This index is not used directly for searching words, as typing and matching the diacritics is cumbersome. However, this index is used to fetch all morphological variations of the query's words during the automatic expansion process. The index contains 19273 distinct words (see Table 1).

The Non-Vowelized Index

This index contains all distinct words of the Quran after they have been processed by stripping their diacritical marks and unifying the alef-hamza character to straight alef (i.e. converting إأآ to ا). The index contains 14918 distinct words. This diacritic-less index is about (22.6%) reduction of the Vowelized-Words Index. The Non-Vowelized Index is considered the main index of the search engine and is used to fetch the query's bag-of-words. So at least one morphological form of each word in the Quran is indexed. The other words are obtained from the Vowelized-Words Index through the query expansion process.

The Root Index

This inverted list contains all distinct roots of the Quran after they have been automatically stemmed and verified against manually extracted roots from the Quran (Khadir 2005). The verification is made for completeness and correctness. A total of 1767 distinct roots (including proper names appearing in the Quran) were identified and referenced in the Vowelized-Words and Non-Vowelized-Words indexes as well as the Thesaurus Index. The results of the stemmer will be discussed in the experimental section. This index is about (88.2%) reduction of the Non-Vowelized Index.

The Thesaurus

It has been argued that Arabic does not have synonyms, especially for the Quran, the way they are used for other languages. Each word in the Quran has been chosen to fit a syntactical and semantical role that cannot be performed by another word. However, in our work we benefited from the work done by (Kubaisi 2006) to collect words from the Quran and group them into semantic classes for the purpose of query expansion.

Stemming

Stemming is the task of correlating several words onto one base form. On average, stemming increases similarities between documents and queries because they have additional common terms after stemming which do not exist before. Stemming has a relatively low processing cost. It uses morphological heuristics to remove affixes from words before indexing. It reduces the index size, and it usually slightly improves results (Strzalkowski and Vauthey 1992). This makes it very attractive for use in IR.

Arabic stemming is more complicated than English stemming. Major words of the Arabic language are

constructed from the three consonant roots by following fixed patterns. Patterns include prefixes, infixes and suffixes to indicate number, gender and tense. Arabic stemming is the process of removing all affixes from a word to extract its root. A stemmer for Arabic, for example, should identify the string, *kateb* كاتب (*writer*), *ketab* كتاب (*book*), *maktabah* مكتبه (*library*), *maktab* مكتب (*office*), as one base form *ktb* كتب (*he wrote*).
In our research, we used a rule-based stemmer (after a kind permission from (Khoja 2002)) to experiment with Arabic passage retrieval and question answering (Hammo et al. 2004). The stemmer performed reasonably with more than (96%) accuracy.

Experimental Design

The Test Collection

For this paper, we used the scripts of the Quran and a collection of forty non-vowelized, words obtained from 10 college students. Each student has been asked to provide 4 words that they memorize from the Quran. The list, without any modification, is given in Table 2.

Table 2. List of words to be Searched

#	Word	#	Word	#	Word	#	Word
1	اثاث	11	السحاب	21	بزغ	31	خشوع
2	اثار	12	السمع	22	بعث	32	زوج
3	اجاج	13	الصافي	23	توبة	33	سابق
4	احب	14	الصبر	24	جدار	34	سباحة
5	الابتر	15	الضعف	25	جلباب	35	ستر
6	الابتسام	16	الفساد	26	جمل	36	طاعة
7	التفكير	17	المجرم	27	جميل	37	ظلم
8	الجن	18	المنهج	28	جوع	38	قبر
9	الرحمة	19	النصر	29	حصاد	39	لهب
10	السجن	20	برهان	30	حظ	40	وفاء

Results form the Tokenizer

We started with one chapter (surah) after being preprocessed as explained earlier. The tokens were manually examined to verify the correctness of the tokenization process. The *Tokenizer* successfully isolated all the words and passed them for the stemmer to extract their roots.

Results from the Stemmer

In this experiment, we run the stemmer against the Non-Vowelized Index and the results were close to (91%) accuracy. We observed that most of the failing cases were due to stemming proper names (such as the names of prophets and names of angels), ancient cities, places and people, numerals, as well as words with doubled characters (represented using the diacritic shada (ّ)). To verify the correctness of the roots, we compared the automatically generated roots with a list of manually extracted and verified roots compiled by (Khadir 2005). We manually corrected the mistaken ones and added the missing ones so the two lists became identical. Finally, the Vowelized and the Non-Vowelized Indexes and the Thesaurus Index have been connected with the Root Index.

Experimenting with the Search Engine

Searching Diacritic-less Words

An Arabic search engine was designed and implemented to perform query processing using standard SQL over the indexing techniques mentioned earlier. Table 3 shows the results of searching the collected 40 non-vowelized words. The system fails in all cases, where an exact match could not be found. Table 4 shows a sample of what is found in the Non-Vowelized-Word Index that did not match with some of tested words. It is clear that the failure in most of the cases is due to the missing diacritics. Figure 3 shows the output of this experiment.

Table 3. Retrieved Verses for the Collected Words

Retrieved Verses (RV)							
Word	RV	Word	RV	Word	RV	Word	RV
اثاث	0	السحاب	2	بزغ	0	خشوع	0
اثار	1	السمع	2	بعث	1	زوج	1
اجاج	1	الصافي	0	توبة	1	سابق	2
احب	3	الصبر	0	جدار	0	سباحة	0
الابتر	1	الضعف	1	جلباب	0	ستر	0
الابتسام	0	الفساد	3	جمل	0	طاعة	1
التفكير	0	المجرم	1	جميل	1	ظلم	3
الجن	3	المنهج	0	جوع	1	قبر	0
الرحمة	3	النصر	1	حصاد	0	لهب	1
السجن	3	برهان	2	حظ	2	وفاء	0

Table 4. Sample of Words Found in the Index

Word	Word in Index
اثاث	اثاثا
اثار	اثارهم، اثارهما
الصبر	صبر، صبرا
بزغ	بازغا
جدار	جدارا
حصاد	حصاده
خشوع	خاشعا
قبر	اقبره

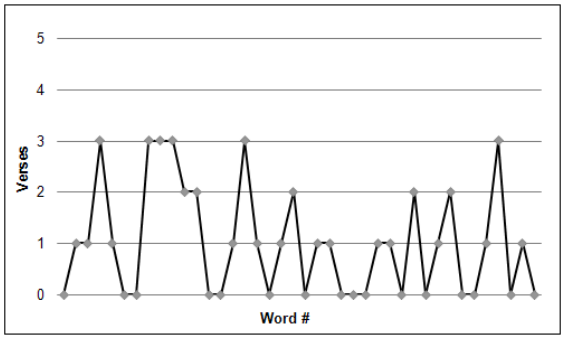

Figure 3. Results of Searching the Collected Words

Effectiveness of Query Expansion

Query Expansion (QE) can be defined as the process of reformulating the query's bag-of-words to overcome the problem of mismatching potential documents and improving the performance of a search engine by including in the results documents that are more relevant (of better quality) , or at least equally relevant (Oiu and Frei 1993; Vectomova and Wang 2006). Without query expansion, the documents which have the potential to be relevant to the user's query would not be retrieved. Many QE techniques have been investigated in the information retrieval literature. They include:

- Query expansion through synonymy. This is performed through finding words synonyms in the search query, and searching for the synonyms as well.
- Query expansion through stemming. This is performed by adding the various morphological forms of the input query, and searching for all forms as well.
- Query expansion through word sensing. This is performed through sensing the words to resolve ambiguity from a specialized database such as the WordNet.
- Query expansion through fixing spelling errors, and automatically searching for the corrected form in the search query.
- Query expansion through paraphrasing. This is performed by rewriting the terms of the original query. Some query expansion techniques such as synonymy and stemming have been criticized for increasing the total recall on the expense of lowering the precision. Other techniques like word sense disambiguation (wsd) tend to increase the precision. However, despite the increase in the recall, augmenting the user's query with synonyms and morphological variations and ranking the occurrences of the query's words, cause documents with more approximate terms to migrate near the top of the ranked list, hence, leading to a higher performance.

Query Expansion through Stemming

The goal of query expansion in this regard is to increase recall at the same time that precision increases. Most of the time using the exact words for searching will not give good results and sometimes no results at all as was shown in Table 3. This is because of the discrepancy in morphological structure between the words in the corpus and the query's words, which most of the time end up with no-matching. We believe that users interested in a word like كتب (writes, 3PSNG/VBD), also are interested in words such as: كتابة (writing), مكتب (office), مكتبة (library), and مكتوب (something written). Nevertheless, all previous words can be correlated to the root (كتب) to be retrieved. Therefore, in our search engine query expansion is done automatically to find all verses with words that correlated to the roots of the query's bag-of-words. The process starts with applying the stemmer on the bag-of-words. For each root we obtain from the query, we search the Root Index and fetch from the Vowelized-Words Index all the diacritic words that are interrelated with this root. The new query of diacritic words is submitted again to search the *Posting* Table for the occurrences of all verses having these words. Running the experiment with expanding the search through stemming was very efficient and satisfactory. Table 5 shows the results of applying the stemmer, while Figure 4 shows the improvement over the 40 words. The obtained results made this technique very practical, especially for a question answering system (Hammo et al. 2004).

Query Expansion through Thesaurus

By expanding a search query to search for the synonyms of a user entered term, the recall is also increased, and sometimes at the expense of precision. When used for information retrieval, terms and their synonyms are augmented in the query vector and a new search begins. We compiled a thesaurus of the Quran words and group them semantically. The words related to the queries and the new findings after expanding the search automatically are given in Table 6, while Figure 5 shows the improvement over the 40 queries.

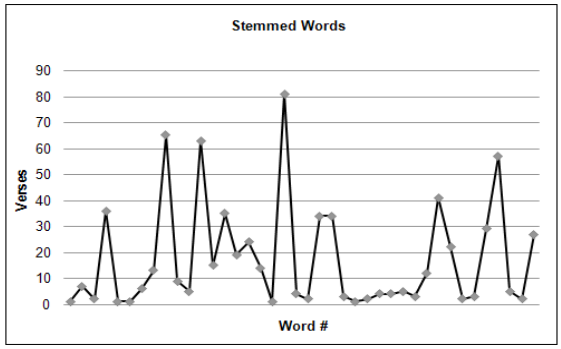

Figure 4. Results of Searching the Quran Using Stemmer

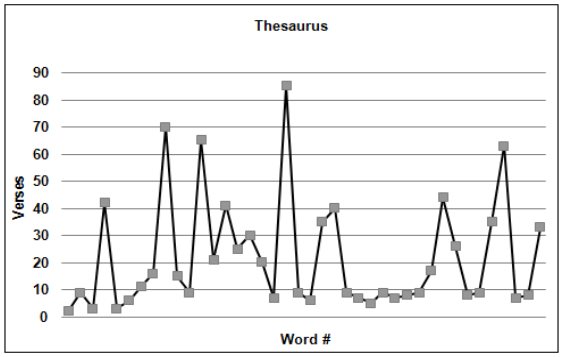

Figure 5. Results of Searching the Quran Using Thesaurus

Table 5. Results of Search Expansion Using the Stemmer

W#	word	verses returned	root	# of distinct words related to this root	# of verses found	W#	word	verses returned	root	# of distinct words related to this root	# of verses found
1	اثاث	0	أثث	1	1	21	بزغ	0	بزغ	2	2
2	اثار	1	أثر	15	7	22	بعث	1	بعث	34	34
3	اجاج	1	أجج	2	2	23	توبة	1	توب	34	34
4	احب	3	حبب	37	36	24	جدار	0	جدر	4	3
5	الابتر	1	بتر	1	1	25	جلباب	0	جلب	2	1
6	الابتسام	0	بسم	1	1	26	جمل	0	جمل	7	2
7	التفكير	0	فكر	6	6	27	جميل	1	جمل	7	4
8	الجن	3	جنن	46	13	28	جوع	1	جوع	4	4
9	الرحمة	3	رحم	65	65	29	حصاد	0	حصد	5	5
10	السجن	3	سجن	9	9	30	حظ	2	حظظ	3	3
11	السحاب	2	سحب	6	5	31	خشوع	0	خشع	12	12
12	السمع	2	سمع	63	63	32	زوج	1	زوج	41	41
13	الصافي	0	صفو	16	15	33	سابق	2	سبق	22	22
14	الصبر	0	صبر	35	35	34	سباحة	0	سبح	31	2
15	الضعف	1	ضعف	34	19	35	ستر	0	ستر	3	3
16	الفساد	3	فسد	24	24	36	طاعة	1	طوع	47	29
17	المجرم	1	جرم	16	14	37	ظلم	3	ظلم	64	57
18	المنهج	0	نهج	1	1	38	قبر	0	قبر	5	5
19	النصر	1	نصر	81	81	39	لهب	1	لهب	2	2
20	برهان	2	برهن	4	4	40	وفاء	0	وفي	42	27

Table 6. Results of Search Expansion using a Thesaurus of Semantic Groups from the Quran

W#	word	root	Semantic Groups						# of verses found
1	اثاث	**أثث**	متاع						2
2	اثار	**أثر**	علامات	اية					9
3	اجاج	**أجج**	ملح						3
4	احب	**حبب**	الشهوة	الهوى	الحب	الشغف	الغرام	الهيام	42
5	الابتر	**بتر**	بتك	قطع					3
6	الابتسام	**بسم**	الضحك	السخرية	الاستهزاء	الازدراء	الاستخفاف		6
7	التفكير	**فكر**	المميز	الوازع	المتدبر	المفكر	الرشيد		11
8	الجن	**جنن**	الابالسة	العفاريت	الشياطين				16
9	الرحمة	**رحم**	البر	التفضل	الانعام	الاحسان	المنة		70
10	السجن	**سجن**	حبس	امساك	توقيف	اثبات	حجر	رباط	15
11	السحاب	**سحب**	الغمام	العارض	الظلة	الصيب			9
12	السمع	**سمع**	انصت	استمع					65
13	الصافي	**صفو**	الزاكي	الطيب	الطاهر	الممحص	المصنوع	المخلص	21
14	الصبر	**صبر**	الحلم	الصوم	العفة	القناعة	كظم	الغيظ	41
15	الضعف	**ضعف**	العجز	الوهن	الوهي	الفتور	الكسل	التثاقل	25
16	الفساد	**فسد**	الخبث	الرجز	الرجس	النجس	القبح	السوء	30
17	المجرم	**جرم**	جبار	قاهر	متكبر	عالي	عتل	عاتي	20
18	المنهج	**نهج**	امام	صراط	طريق	سبيل	فج	جدد	7
19	النصر	**نصر**	الظفر	الغلبة	الفتح	الفوز			85
20	برهان	**برهن**	بينة	اية	الاء	حجة	بصائر		9
21	بزغ	**بزغ**	بدا	ظهر	طلع	برز			6
22	بعث	**بعث**	ارسل						35
23	توبة	**توب**	اناب	اب	اسف	رجع	انتهى	فاء	40
24	جدار	**جدر**	حائط	سور	سد	ردم	برزخ	حاجز	9
25	جلباب	**جلب**	ثوب	لباس	كساء	خمار	سربال	ريش	7
26	جمل	**جمل**	ابل	بعير	ناقة				5
27	جميل	**جمل**	جمال	حسن	وسامة	زينة	نضارة		9
28	جوع	**جوع**	خصاصة	مخمصة	مسغبة				7
29	حصاد	**حصد**	جنى	خضد	قطف				8
30	حظ	**حظظ**	بعض	جزء	قسم	نصيب	كفل	قط	9
31	خشوع	**خشع**	اخبات	تضرع	اطمئنان	لين	خضوع		17
32	زوج	**زوج**	بعل	سيد	صاحب				44
33	سابق	**سبق**	جاوز	عبر	قطع	سارع			26
34	سباحة	**سبح**	جرى	مر	ركض	نفر	زحف	انطلق	8
35	ستر	**ستر**	بطن	اخفى	يعزب	اسر	كتم	حجب	9
36	طاعة	**طوع**	الاذعان	الذل	الخضوع	الاستسلام	الاستكانة	الاخبات	35
37	ظلم	**ظلم**	بغي	عدوان	طغيان	جور	حيف	جنف	63
38	قبر	**قبر**	جدث	لحد					7
39	لهب	**لهب**	جذوة	شعلة	شهاب	قبس	شرر	شواظ	8
40	وفاء	**وفي**	اجر	ثمن	عطاء	جزاء	ثواب	فضل	33

Conclusion

In this paper, we explained the problem of searching diacritized text and provide a solution for the searching problem through indexing. We investigated the use of query expansion on searching the Quran in the absence of diacritics, where queries are automatically augmented with related terms extracted from a vowelized index by applying a stemmer and a thesaurus of semantic classes. We conducted a set of experiments on searching words from the Quran and we concluded that query expansion for searching Arabic text is promising and it is likely that the efficiency can be further improved. Figure 6 shows a comparison of applying the different techniques on the 40 queries. In the future, we plan to use these ideas in improving a question answering system for the Arabic language.

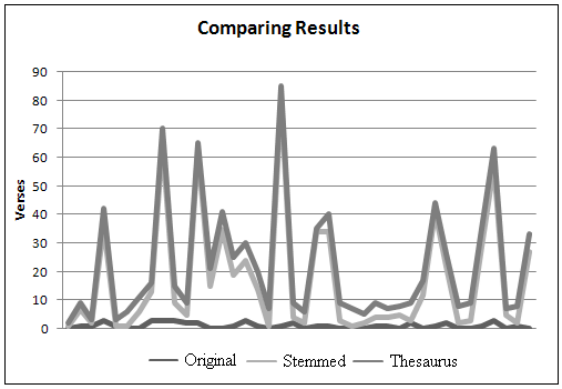

Figure 6. Comparison of the Searching Techniques

References

Abu-Salem H.; Al-Omari M.; and Evens M. 1999. Stemming Methodologies over Individual Query Words for an Arabic Retrieval System. Journal of the American Society for Information Science (JASIS) 50(6): 524-529.

Alsamara K.; Abu-Salem H.; Abuleil S.; and Hammo B. 2003. Building Automated Indexes to Improve Arabic Information Retrieval. Proceedings of 2003 International Conference on Management Science & Engineering, 2573-78. Georgia, USA.

Buckwalter T. 2007. Issues in Arabic Morphological Analysis. In A. Soudi, A. Van den Bsch, and Gunter Neumann, eds., Arabic Computational Morphology, 23-41. Berlin, Springer-Verlag.

Debili F.; Achour H.; and Souissi E. 2002. De l'etiquetage grammatical a' la voyellation automatique de l'arabe. Technical report, Correspondances de l'Institut de Recherche sur le Maghreb Contemporain 17.

El-Sadany T., and Hashish M. 1988. Semi-automatic vowelization of Arabic verbs. In 10th NC Conference, Jeddah, Saudi Arabia.

Emam O., and Fisher V. 2004. A Hierarchical Approach for the Statistical Vowelization of Arabic Text. Technical report, IBM patent filed, DE9-2004-0006, US patent application US2005/0192809 A1.

Gal Y. 2002. An HMM approach to vowel restoration in Arabic and Hebrew. In ACL-02 Workshop on Computational Approaches to Semitic Languages, 27-33.

Hammo, B.; Abu-Salem, H.; Lytinen, S.; and Evens, M. 2002. QARAB: A Question Answering System to Support the Arabic Language. Workshop on Computational Approaches to Semitic Languages, 55-65. Philadelphia, PA.

Hammo, B.; Abuleil, S.; Lytinen, S.; and Evens, M. 2004. Experimenting with a Question Answering System for the Arabic Language. Journal of Computers and the Humanities. 38 (4): 379-415.

Hmeidi, I.; Kanaan, G.; and Evens, M. 1997. Design and Implementation of Automatic Indexing for Information Retrieval with Arabic Documents. Journal of the American Society for Information Science. 48 (10): 867-881.

Khadir M. 2005. Quran lexicon. Available at http://www.almishkat.com/words/book.htm

Khoja, S. 2001. APT: Arabic Part-of-Speech Tagger. In Proceedings of the Student Workshop at NAACL 2001, 20-25. Stroudsburg, PA: ACL Press.

Kirchhoff K., and Vergyri D. 2005. Cross-dialectal data sharing for acoustic modeling in Arabic speech recognition. Speech Communication, 46(1):37–51.

Kubaisi A. 2006. Quran words. Available at http://www.islamiyyat.com/kalema.htm.

Lundquist, C.; Frieder, O.; Holmes, D.; and Grossman, D. 1997. A Parallel Relational Database Management System Approach to Relevance Feedback in Information Retrieval. Journal of the American Society of Information Science (JASIS), 50 (5): 413-426.

Qiu Y., and Frei H. 1993. Concept Based Query Expansion. In Proceedings of SIGIR-93, 16th ACM International Conference on Research and Development in Information Retrieval, Pittsburgh, SIGIR Forum, ACM Press.

Salton, G., and McGill, M. 1983. Introduction to Modern Information Retrieval. McGraw-Hill Book Company, New York.

Salton, G. 1989. Automatic Text Processing -The Transformation Analysis and Retrieval of Information by Computer. Addison Wesley, MA.

Strzalkowski, T., and Vauthey, B. 1992. Information Retrieval Using Robust Natural Language Processing. In Proceedings of ACL-92, 104–111.

Vectomova O., and Wang Y. 2006. A study of the effect of term proximity on query expansion. Journal of Information Science 32 (4): 324–333.

Zitouni I.; Sorensen J.; Luo X.; and Florian R. 2005. The Impact of Morphological Stemming on Arabic Mention Detection and Coreference Resolution. In Proceedings of the ACL Workshop on Computational Approaches to Semitic Languages, 63–70. Ann Arbor, June.

Zitouni I.; Sorensen J.; and Sarikaya R. 2006. Maximum Entropy Based Restoration of Arabic Diacritics. Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the ACL, 577–584. Sydney.