Towards an Error-Free Arabic Stemming

Eiman Tamah Al-Shammari
George Mason University
Dept. of Computer Science
4400 University Drive
Fairfax, VA 22030
Eiman.tamah@gmail.com

Jessica Lin, Ph.D.
George Mason University
Dept. of Computer Science
4400 University Drive
Fairfax, VA 22030
jessica@cs.gmu.edu

ABSTRACT
Stemming is a computational process for reducing words to their roots (or stems). It can be classified as a recall-enhancing or precision-enhancing component.

Existing Arabic stemmers suffer from high stemming error-rates. Arabic stemmers blindly stem all the words and perform poorly especially with compound words, nouns and foreign Arabized words.

The Educated Text Stemmer (ETS) is presented in this paper. ETS is a dictionary free, simple, and highly effective Arabic stemming algorithm that can reduce stemming errors in addition to decreasing computational time and data storage.

The novelty of the work arises from the use of neglected Arabic stop-words. These stop-words can be highly important and can provide a significant improvement to processing Arabic documents.

The ETS stemmer is evaluated by comparison with output from human generated stemming and the stemming weight technique.

Categories and Subject Descriptors
H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing – *Indexing methods, Linguistic processing*.

General Terms
Algorithms, Documentation, Experimentation, Human Factors, Languages, Standardization.

Keywords
Arabic, Lemmatization, Stemming, Text Mining, Tokenization.

1. INTRODUCTION
Stemmers are basic elements in query systems, indexing, web search engines and information retrieval systems (IRS). Stemming offers the benefits of minimizing storage requirements by eliminating redundant terms, as well as increasing matching probability for document comparison and unifying vocabulary [1].

iNEWS'08, October 30, 2008, Napa Valley, California, USA.

Unfortunately, stemming can cause errors in the form of over-stemming, mis-stemming and under-stemming. These errors decrease the effectiveness of stemming algorithms [2] however reducing one type of errors can lead to an increase of the other [3].

Over-stemming occurs when two words with different stems are stemmed to the same root. An over-stemming example is when the word "probe" and "probable" are merged together after stemming.

Under-stemming occurs when two words that should be stemmed to the same root are not, for example, when the stemmer fails to conflate the words "adhere" and the word "adhesion" to the same root.

Mis-stemming is defined as "taking off what looks like an ending, but is really part of the stem [4] for example, stemming the word "red to "r" or the word "reply to "rep".

The challenges associated with stemming are even more pronounced in Arabic. Arabic is one of the most complex languages, in both its spoken and written forms. However, it is also one of the most common languages in the world. The Arabic language exhibits a very complicated morphological structure.

This paper presents a stemming algorithm that relies on Arabic language morphology and Arabic language syntax. The stemming algorithm automatically identifies nouns and verbs without the need for a dictionary. Nouns and verbs are stored in a separate dictionary. Automated addition to the syntactic knowledge and construction of corpus-based dictionaries reduce both stemming errors and stemming cost.

The remainder of this paper is organized as follows:

Section II includes a brief review of Arabic language morphology and discusses previous Arabic language stemming processes. Section III introduces the proposed methodology followed by a description of the stemming algorithm in section IV. Section V presents the evaluation criteria and experimental results. A conclusion and discussion of future work can be found in section VI.

2. BACKGROUND AND RELATED WORK
Arabic language is a semantic language with a composite morphology. Arabic words are categorized as particles, nouns, or verbs [5].

Unlike most western languages, Arabic script writing orientation is from right to left. There are 28 characters in Arabic. The characters are connected and do not start with capital letter as in English. Figure 1 below shows a list of Arabic characters. Furthermore, most of the characters differ in shape based in their position in the sentence and adjunct letters. Figure 2 below demonstrates some of the Arabic characters changes in shape.

خ	ح	ج	ث	ت	ب	ا
ص	ش	س	ز	ر	ذ	د
ق	ف	غ	ع	ظ	ط	ض
ي	و	ه	ن	م	ل	ك

Figure 1: Arabic characters (letters)

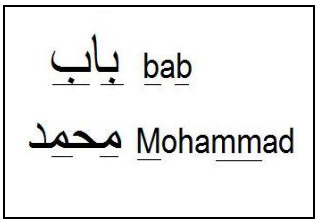

Figure 2: Arabic characters shape differ as their position in the word change

In Arabic, proper nouns do not start with capital letter as in English, which makes one particularly challenging task for machines, recognizing and extracting proper nouns from Arabic texts.

Furthermore, in English, words are formed by attaching prefixes and suffixes to either or both sides of the root. For example the word *Untouchables* is formed as follows

Un	*touch*	*able*	*s*
Prefix	Root	First Suffix	Second Suffix

In Arabic, additions to the root can be within the root (not only on the word sides) which is called an infix. This causes a serious issue in stemming Arabic documents because it is hard to differentiate between root characters (letters) and affix letters. For example, for the root "drink" **شرب** in Arabic, adding the infix letter "ا" (circulated in Figure 3) formed a different word: **شارب** "drinker".

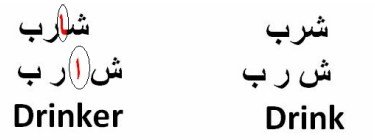

Figure 3. Arabic infix example

Table 1 displays an example of the Arabic Word = الشارب (drinker) and its stems with the common prefixes and suffixes.

Table 1. Arabic Example

Prefixes + Stem (Root + Pattern) + Suffixes		
Root	شرب	drink
Prefixes	ال	the
Stem	شارب	drinker
Suffixes	ين OR ان	dual
Suffixes	ون	plural
Suffixes	ة	feminine
الشاربان	the drinkers (dual)	
الشاربين	the drinkers (plural)	
الشارب	the drinker (masculine)	
الشاربه	the drinker (feminine)	

Suffixes, prefixes and infixes are categorized based on their uses. Similar to other Western languages, there are specific suffixes to convert the word from the singular form to the plural form and others to convert from masculine to feminine.

Due to its complicated morphological structure, Arabic requires a different stemming process from other languages.

Automatic Arabic stemming proved to be an effective technique for text processing for small collections [6-8] and large collections [9,10] of documents. Xu et al. [11] showed that spelling normalization combined with the use of tri-grams and stemming could significantly improve the accuracy of Arabic text processing by 40%. Additionally, in Al-Shammari et.al [6] it was proven that stemming can improve text clustering.

Stemming Arabic documents was performed manually prior to TREC (Text Retrieval Conference) and only applied on small corpora. Later, many researchers both native and non-native Arabic speakers created a considerable amount of Arabic stemming Algorithms.

Based on the required level of analysis, Arabic stemmers are categorized as either root-based [12, 13] or stem-based [9, 10, 14] .

In Arabic, the root is the original form of the word before any transformation process [15]. However, a stem is a morpheme or a set of concatenated morphemes that can accept an affix [16].

A superior root-based stemmer is the Khoja's stemmer[14], presented by Khoja and Garside[12] . The Khoja algorithm removes suffixes, infixes and prefixes and uses pattern matching to extract the roots. The algorithm suffered from problems especially with names and nouns.

A possible solution for this problem is to add a lookup dictionary to check the nouns, roots and names. Although this solution seems straightforward and easy, this process is computationally expensive. Al-Fedaghi and Al-Anzi [17] estimated that there are around 10,000 independent roots. Each root word can have prefixes, suffixes, infixes, and regular and irregular tenses.

On the other hand, there have been several proposed Arabic stem-based (light) algorithms [9, 10, 14, 18-20]. The prominent Arabic light stemmer is Aljlayl [14, 18] light stemmer. Light stemming

does not deal with patterns or infixes; it is simply the process of stripping off prefixes and/or suffixes. Unfortunately, the unguided removal of a fixed set of prefixes and suffixes causes many stemming errors especially where it is hard to distinguish between an extra letter and a root letter.

Although light stemmers produce fewer errors than aggressive root-based stemmers; in contrast, aggressive stemmers reduce the size of the corpus significantly. Paice [3,21,22] proved that light stemming reduces the over-stemming errors, but increases the under-stemming errors. On the other hand, heavy stemmers reduce the under-stemming errors while increasing the over-stemming errors.

Both Arabic root-based and stem-based algorithms suffer from generating stemming errors. The main cause of this problem is the stemmer's lack of knowledge of the word's lexical category (i.e. noun, verb, proposition, etc.)

To mitigate the drawbacks of the previous work on Arabic stemming, we propose an alternative that defines a rule to stem words instead of chopping off the letters. This rule is set by the syntactical structure of the word. For example, verbs require aggressive stemming and need to be represented by their roots. Nouns on the contrary only require light suffixes and prefixes elimination. This advanced stemming is known as Lemmatization [6].

In this work, we propose the first Arabic stemming algorithm that uses the syntactical knowledge to make stemming decisions, and we hypothesize that my approach will be more efficient in tokenizing Arabic documents than the existing approaches. In addition to the general stemming benefits, my approach can reduce the stemming errors, as well as stemming cost by reducing unnecessary stemming. An earlier version of my stemmer was introduced in and proved to improve clustering [6].

3. METHODOLOGY

Stop words (functional words or structural word list [1]) are words that either carry no meaning or are very common [23] thus do not represent the document. Stop words list usually contains prepositions, pronouns, and conjunctions.

Text processing often performs stop words removal early in the process, although there is currently no standardized list of Arabic Stop Words. The current available Arabic stop words list [9] introduces less than 200 words.

we was able to define more than 2,200 stop words and categorize them into "useful" and "useless" stop words.

Useless stop words are stop words that are used extensively and give no benefits to the subsequent words. On the contrary, useful stop words are words that can indicate the syntactical categories of the subsequent words. For example, in an English sentence such as *"I read a book yesterday,"* it is easy to realize that book is a noun and thus does not require aggressive stemming. Table 2 and 3 are examples of useful stop words.

Unfortunately, due to the early removal of the stop words, this valuable information is lost. The same scenario applies to Arabic language too. we believe that the useful stop words can help us identify nouns and verbs and direct us into the appropriate stemming. ETS automatically identifies nouns and verbs and generates global nouns and verbs dictionaries. The benefit of these dictionaries is to find similar nouns in the corpus that were used differently in other sentences. For example, in the following paragraph the word book is identified as a noun and was recognized as a noun in the following sentence.

I read a book yesterday, I love books.

In Table 2, a sub list of stop words preceding verbs is shown, and Table 3 presents some of the stop words preceding nouns. The stop words list was initially generated by three methods; English stop words translation, identification of common words in arbitrary Arabic documents, and manual search of synonyms to the previously identified stop words.

In the following section the ETS algorithm will be described in detail.

Table 2. Preposition Preceding Verbs

Preposition	English
حيثما	Wherever
كلّما	Whenever
إذا	If
عندما	When (not for question)

Table 3. Arabic circumstantial nouns indicating time and place

Preposition	English Equivalence
إلى	until ,near, towards ,to
أمامَ	in front of
باتجاه	On the direction of
بجانب	Aside, next to, beside
بعد	After
بينَ	Between
تحت	Below, beneath, down
حتى	Till (time and location)
خارج	Outside of
خلال	Through, during,
عبر	Through
على	Over
فوق	Above, up
في	In (time, location, duration)
قبلَ	Before
قريب	Near
منذ	since
وراءَ	Behind ,Beyond

4. ARABIC STEMMING ALGORITHM

Prior to applying the ETS algorithm, normalization is performed to make the data sets more consistent. Normalization consists of the following steps:

- Convert text to Unicode.
- Remove diacritics and punctuation.
- Remove non letters (for example, numbers).
- Replace آ with double alif ا.
- Replace ى with ا .
- Replace initial إ with أ.
- Replace all hamza forms ئ , ؤ , ء with أ

As shown in Figure 4, the algorithm consists of different phases. During the first phase, useless stop words are removed to reduce the size of the corpus. Next, nouns are identified by either locating stop words that always precede nouns (example: the, a, over, etc) or words starting with definite articles. At this level, these words are flagged as nouns as a preparation for the stemming phase. In parallel to that process verbs are found by locating stop words that always precede verbs. Similar to the nouns, the verbs are added to a global verb dictionary and tagged as verbs.

In Arabic, we cannot have two consecutive verbs, thus any word following a verb is either a stop word or a noun. If the word is not a stop word then the word is added to the noun dictionary and flagged as a noun. Verbs are stemmed using the Khoja root-based stemmer and nouns are lightly stemmed. The new stemmed nouns and verbs are also added to the nouns and verb dictionary respectively.

The document is revisited by categorizing words with missing flags using the noun corpus and the verb corpus as a lookup table. Nouns usually co-occur in the same document, thus the lookup table will allow us to identify un-flagged nouns. Other words that do not belong in any category will be treated as nouns and stemmed lightly. Before we direct a word to the appropriate stemming by the word flag, all the stop words are removed since they offer no further advantage. Table 4 below summarizes the algorithm.

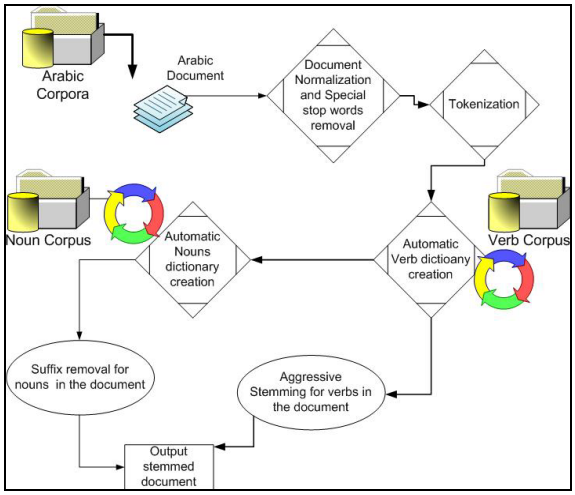

Figure 4. The educated Arabic stemming algorithm simplified

Table 4. The Educated Text Stemmer (ETS) Arabic Stemming Algorithm

ETS
Input: Arabic document **Output**: Stemmed document. Noun Dictionary. Verbs Dictionary. **V**: Verb dictionary (one dimensional array sorted alphabetically[1]) **N**: Noun dictionary (one dimensional array sorted alphabetically) **NSW**: Array of stop words proceeding nouns **VSW**: Array of stop words proceeding verbs **SW**: Array of stop words (including both NSW and VSW)
1. Remove useless stop words 2. Locate words attached to definite articles, and preceded by NSW and flag them as Nouns 3. Add nouns to the noun dictionary N. 4. Locate Verbs proceeded by VSW. Flag verbs in the document. 5. Add the identified Verbs to the verb dictionary V. 6. Revisit the document searching for nouns and verbs existing in the document. 7. Tokens (words) with missing tags are treated as nouns. 8. Remove the rest of the stop words (useful stop words). 9. Apply light stemming Algorithm on nouns. 10. Apply Khoja's root-based stemmer on verbs.

5. EVALUATION AND EXPERIMENTS

Different criteria are used to evaluate the performance of a stemmer. A good stemmer (by definition) is a stemmer that stems all the words to their correct roots.

Paice [3,22] introduced The stemming weight (SW) as an indicator to the stemmer efficiency. Stemming weight is the ratio between the under stemming errors and the over stemming errors.

For text mining applications that deal with a massive number of documents, the ability to significantly reduce the size of the text is also a desirable property for a good stemmer. In my previous work [6] we compared the effect of my algorithm on document clustering to that of the leading Arabic root-based stemmer presented by Khoja using Cluster Purity [24]. Applying K-means clustering on the three datasets leads to an overall cluster purity of 70.8% for the documents stemmed by ETS and 58% for the documents stemmed by Khoja's stemmer.

[1] For fast lookup, these dictionaries can be implemented using hash tables

In summary, measures discussed in the literature [21, 22, 25] to evaluate the performance of a stemmer include algorithm speed, storage saving (compression), stemming weight and retrieval effectiveness.

5.1 Paice's Evaluation Methodology

The first evaluation approach is an evaluation method proposed by Paice, applied to compare various English stemmers.

Paice introduced three new quantitative parameters to evaluate a stemmer performance: Under-stemming index (UI) , Over-stemming index(OI), and their ratio , the stemming weight (SW).

A group of morphologically and semantically related words are submitted to the stemmer. If the stemmer produces more than one stem (root for Arabic) for the same group then the stemmer has made an under-stemming error. If words belonging to different groups are stemmed to the same stem then the stemmer has made an over-stemming error. The ideal stemmer should be able to conflate (group) the related words to the same stem and has low UI and OI.

The Khoja stemmer (root-based) tends to stem morphologically related words (but not necessarily semantically related) and as a result has a high over-stemming error rate.

Arabic light stemmers only remove frequent suffixes and prefixes from the word leading to a high under-stemming error rate. To achieve a balance between these two parameters, the ratio between UI and OI, Stemming Weight (SW) is introduced.

To generate the samples to test the stemmer, we have to generate different words from a single stem by adding suffixes, infixes and prefixes. Contrary to English, in Arabic, adding affixes, prefixes or suffixes to the word can change the meaning completely. Therefore, creating a group of morphologically and semantically related words is not a trivial task. To achieve this, we first created all the possible derivations of a single stem, and then eliminated the word that does not belong semantically.

Suppose we had the following samples divided into two groups as shown in Table 5. Group 1 represents the root child and its derivations and group two represents the word parasite and it is derivations.

Table 5 : List of words derived from the stem طفل

Group 1	طفل	Child
	أطفال	Children
	الأطفال	The Children
	طفلكم	Your child
	أطفالكم	Your children
	طفولة	Childhood
	للطفولة	For childhood
Group 2	طفيلي	parasite
	طفيليات	parasites
	طفيل	parasite

The Desired merge total (DMT) is the number of different possible word form pairs in the particular group, and is given by the formula: [22,26]

$$DMT = 0.5\ n\ (n - 1)$$

Where n is the number of words in that group.

The Global Desired Merge (GDMT) is the sum of all the DMT's of the various samples.

There exists a case where certain words in a specific group can be conflated after stemming with words from another semantic group. To count all the possible word pairs formed we use the Desired Non-merge Total (DNT):

$$DNT = 0.5\ n\ (W - 1)$$

Where W is the total number of words, the sum DNT for all the groups is defined as the Global Desired Non-Merge (GDNT).

After the stemming process is performed, we would like to see if all the words in a group are conflated in the same group. To quantify the stemmer's inability to merge these words, Paice introduced the "Unachieved Merge Total" (UMT):

$$UMT = 0.5 \sum_{i=1}^{s} u_i (n - u_i)$$

Where s is the number of distinct stems, and ui is the number of instances of each stem.

From the sum of UMT for each group (in our example 2 groups), we obtain the Global Unachieved Merge Total (GUMT).

The under-stemming index (UI) is: GUMT/GDMT [27].

Table 6 displays the output of both stemmers. In Tables 7 and 8 the output of the ETS stemmer and the Khoja stemmer for group one and group two respectively is demonstrated.

Table 6: A comparison between the stemmers output for Sample1, the right stem is underlined.

Word	Khoja output	ETS output
طفل	طفل	طفل
أطفال	طفل	طفل
الأطفال	طفل	طفل
طفلكم	طفل	طفل
أطفالكم	طفل	طفل
طفولة	طفل	طفل
للطفولة	طفل	طفل
طفيلي	طفل	طفيل
طفيليات	طفل	طفيل
طفيل	طفل	طفيل

A stemmer might transform different words to the same stem (over-stemming). For such cases Paice introduced the Wrongly Merged Total (WMT), which is the count of over-stemming errors for each group[27].

$$WMT = 0.5\sum_{i=1}^{t} v_i(n_s - v_i)$$

Where t is the number of original groups that share the same stem, n_s is the number of instances of that stem, and, v_i is the number of stems for group t.

Similar to the above we can obtain the Global Wrongly Merged Total (GWMT) by summing the WMT for all the groups.

The over-stemming index (OI) is: GWMT/GDNT.

The table below demonstrates the evaluation of both stemmers on both groups.

Table 7: Khoja stemmer evaluation

	DMT	DNT	UMT	WMT
	24	45	0	10.5
Totals	46	139	0	10.5

Table 8: ETS stemmer evaluation

	DMT	DNT	UMT	WMT
	24	45	0	0
Totals	46	139	0	0

From the previous experiment, Khoja's stemmer had no under-stemming errors but had a UI = 0.0755. For the ETS stemmer, both UI and OI were zero.

The Khoja stemmer is an aggressive stemmer, therefore the under-stemming error was expected. We continued the experiments on many other samples, in all the cases the ETS were able to make less over-stemming errors than the Khoja stemmer. We can conclude that my stemmer is more efficient than the Khoja stemmer.

5.2 Comparison to an Expected Output

In order to assess the effectiveness of the stemmer, we compared its output to that of a manually generated stem. we randomly picked two samples of Arabic documents. The first sample contains 47 medical documents (total of 9435 words), and the second sample contains 10 long, Arabic sports articles from CNN.com (total of 7071 words).

The samples were processed manually, and words were assigned to their correct stem. In addition, a noun and a verb dictionary were created by the tester (Native Arabic speaker).

After manually generating the set of expected output, we ran my stemmer on the same set of samples. We performed these experiments at the designing phase of the ETS stemmer, thus we was able to analyze the output of the ETS stemmer and tackle the errors. we recursively repeated the experiments after adding extra rules and additional stop words.

At the end of the testing phase, on average, the ETS stemmer was able to generate 96% correct stems.

We observed that the ETS stemmer produces better results when more documents are involved in the stemming process. The automatic generation of global nouns and verbs dictionaries helps to identify nouns and verbs appearing in different context. The system initializes a new noun and verb dictionary for every experiment. The storage and the reuse of dictionaries will improve the accuracy of the stemmer where more nouns and verbs are recognized.

6. CONCLUSION AND FUTURE WORK

In this paper we introduced a novel approach for stemming Arabic documentation. we compared the performance of the new algorithm with that of the Khoja stemming algorithm using stemming weight as an evaluation criterion.

We showed that the use of presently neglected Arabic stop words can be highly effective and can provide a significant improvement when processing Arabic documents.

Additionally we introduce a new framework to normalize Arabic documents by overcoming the limitations of previous approaches, caused by the early removal of stop words.

The experiments showed a promising future for the stemming approach, which encourages further research into more in-depth comparisons of its performance with that of other leading stemmers.

7. REFERENCES

[1] G. Salton, *Automatic text processing: the transformation, analysis, and retrieval of information by computer*, Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 1989.

[2] R.A. Baeza-Yates, "Text-Retrieval: Theory and Practice," North-Holland Publishing Co., 1992, pp. 465-476.

[3] C.D. Paice, "An evaluation method for stemming algorithms," Dublin, Ireland: Springer-Verlag New York, Inc., 1994, pp. 42-50.

[4] "Snowball: A language for stemming algorithms"; http://snowball.tartarus.org/texts/introduction.html.

[5] M. Al-Saeedi, "Awdah Almasalik ila Alfiyat Ibn Malek," *Published by Dar ihyaa al oloom {In Arabic}. Beruit, Saudi Arabia*, 1999.

[6] Eiman Al-Shammari and Jessica Lin, "A novel Arabic lemmatization algorithm," *Proceedings of the second workshop on Analytics for noisy unstructured text data*, 2008, pp. 113-118.

[7] I.A. Al-Kharashi, "Micro-AIRS: A microcomputer-based Arabic information retrieval system comparing words, stems, and roots as index terms," 1991.

[8] I.A. Al-Kharashi and M.W. Evens, "Comparing Words, Stems, and Roots as Index Terms in an Arabic Information

Retrieval System.," *Journal of the American Society for Information Science*, vol. 45, 1994, pp. 548-60.

[9] L.S. Larkey and M.E. Connell, "Arabic Information Retrieval at UMass in TREC-10," *Proceedings of the Tenth Text REtrieval Conference (TREC-10)", EM Voorhees and DK Harman ed*, 2001, pp. 562-570.

[10] L.S. Larkey, L. Ballesteros, and M.E. Connell, "Improving stemming for Arabic information retrieval: light stemming and co-occurrence analysis," Tampere, Finland: ACM, 2002, pp. 275-282.

[11] J. Xu, A. Fraser, and R. Weischedel, "Empirical studies in strategies for Arabic retrieval," *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, 2002, pp. 269-274.

[12] S. Khoja and R. Garside, "Stemming Arabic Text," *Lancaster, UK, Computing Department, Lancaster University*, 1999.

[13] W. Al-Fares, "Arabic root-based clustering: An algorithm for identifying roots based on n-grams and morphological similarity," 2002.

[14] M. Aljlayl and O. Frieder, "On arabic search: improving the retrieval effectiveness via a light stemming approach," *Proceedings of the eleventh international conference on Information and knowledge management*, McLean, Virginia, USA: ACM, 2002, pp. 340-347.

[15] M. George, *Al Khaleel: A dictionary of Arabic syntax terms*, Beirut: Library of Lebanon, 1990.

[16] M.A. Al Khuli, "A Dictionary of theoretical linguistics: English-Arabic with an Arabic-English glossary," 1982.

[17] S.S. Al-Fedaghi and F. Al-Anzi, "A New Algorithm to Generate Arabic Root-Pattern Forms," *Proceedings of the 11th National Computer Conference and Exhibition*, 1989, pp. 391–400.

[18] M.A. Aljlayl, "On Arabic Search: The Effectiveness of Monolingual and Bidirectional Information Retrieval," 2002.

[19] H.K. Al Ameed et al., "Arabic Light stemmer: a new Enhanced Approach."

[20] K. Darwish, *Al-stem: A light Arabic stemmer*, 2002.

[21] C.D. Paice, "Another stemmer," *SIGIR Forum*, vol. 24, 1990, pp. 56-61.

[22] C.D. Paice, "Method for evaluation of stemming algorithms based on error counting," *Journal of the American Society for Information Science*, vol. 47, 1996, pp. 632-649.

[23] K. Lin and R. Kondadadi, "A Word-Based Soft Clustering Algorithm for Documents," *Proceedings of 16th International Conference on Computers and Their Applications*, 2001.

[24] Y. Zhao and G. Karypis, "Criterion Functions for Document Clustering," *Experiments and Analysis University of Minnesota, Department of Computer Science/Army HPC Research Center*.

[25] W.B. Frakes, "Stemming algorithms," 1992.

[26] C.D. Pake, "An Evaluation Method for Stemming Algorithms," *SIGIR'94: Proceedings of the 17th Annual International ACM-Sigir Conference on Research and Development in Information Retrieval, Dublin, Ireland, July 1994*, 1994.

[27] V.M. Orengo and C. Huyck, "A stemming algorithm for the portuguese language," *String Processing and Information Retrieval, 2001. SPIRE 2001. Proceedings. Eighth International Symposium on*, 2001, pp. 186-193.