Enhancement Tools for Arabic Web Search

A Statistical Approach

Adnan H. Yahya
Department of Computer Systems Engineering
Birzeit University
Birzeit, Palestine
yahya@birzeit.edu

Ali Y. Salhi
Department of Computer Systems Engineering
Birzeit University
Birzeit, Palestine
asalhi@birzeit.edu

Abstract— **The Arabic web content is growing rapidly and the need for its efficient management is gaining importance and the morphological complexity of Arabic raises many challenges in this regard. This paper reports on some of our work aimed at designing text mining and query pre-processing tools that are able to efficiently process and search large quantities of Arabic web data. In our research we try to address the challenges Arabic poses for natural language processing (NLP) and information retrieval: root extraction, language detection, and Arabic query correction, suggestion and expansion. While not reported in detail here, we are also developing tools for automatic Arabic document categorization. All through, we employ a statistical/Corpus-based approach based on data obtained from a variety of sources. Based on corpus statistics we constructed databases of words and their frequencies as single, double and triple expressions and used that as the infrastructure for the well structured search aid tools that are able to handle the sophisticated nature of Arabic, and capable of being integrated into existing web search engines and document processing systems. We also utilize context analysis and spellchecking of the user queries to enable a more complete and efficient search. While the results reported here are promising, they must be viewed as work in progress, still in need of testing, refining, integration and deployment in real life settings.**

Keywords-component; Natural Language Processing; Information retrieval; Root extraction; Language detection; Arabic query correction

I. INTRODUCTION

As the World-Wide Web (Web) rapidly expands, structured information retrieval systems that help find and manage needed information efficiently acquire added importance. That explains the growing influence of search engine companies. The estimates of the current size of the Web vary from 15 to 30 billion pages [1]. It is a real challenge to deal with this volume, and studies show that the growth rate of the Web is large and sustained, also because many existing pages are being continuously updated. The share of the Arabic language is around 1.4% of the Web total pages [2]. Despite this small share, retrieving Arabic information seems to be an annoying and unsatisfying experience for many users.

The main focus of this paper is to report on our work aimed at designing text mining and query pre-processing tools that are able to efficiently process and search large quantities of Arabic web data. In our research we try to address some of the challenges Arabic poses for NLP and information retrieval: root extraction, language detection, and Arabic query correction, suggestion and expansion. All through, we employ a statistical/corpus-based approach based on contemporary data obtained from a large variety of sources. Based on corpus statistics we constructed databases that have Arabic words with their frequencies and used that as basis to create well structured search aid tools that are able to handle the sophisticated nature of the Arabic language and which are capable of being integrated into existing web search engines and document processing systems. Additionally, we utilize context analysis and spellchecking to enable a more complete and efficient response to user queries.

II. BACKGROUND

A. *Search Engines, Historical Review*

Going back to the history of Web search, we see that the first tool used for searching the Internet was Archie. It was created in 1990 by Alan Emtage. The first Web search engine was Wandex, with indexes collected by the World Wide Web Wanderer, a Web crawler developed by Matthew Gray in 1993. Lycos began in the spring of 1994; Yahoo became available in the same year. NCSA Mosaic in 1993 and Netscape in 1994[3] and in 1998 Larry Page and Sergey Brin came up with Google; a revolution to the search engine concept with all the new ideas, algorithms and visions that came with it.

B. *Search Engine Structure*

Web search engine technology consists of crawling strategies, storage, indexing, ranking techniques and Query engine [4]. However, our focus in this paper is on the enhancements that need to be made for query pre-processing using tools that enable efficient search in large quantities of Arabic web data.

C. *Helping Search Engines Understand What Users Want*

The ambiguity in words/phrases is less of an obstacle when it comes to human beings; they have many communication tools that help remove this ambiguity. Things are more complicated for machines. Search engines have to understand the user intention so as to provide satisfactory results sought by that user. The search engine can achieve that through a

multiplicity of mechanisms such as Query Suggestion, Cross Language Query Suggestion (Suggestion and Translation to different languages), Web and Query Categorization and Query Analysis and NLP tools aimed at enhancing the user experience when performing a search by providing language support as spell checking, auto suggestions and query expansion, language detection, proper name correction and others. In the following sections we will discuss some of our work on developing such Query Analysis/NLP tools.

III. ARABIC RESOURCES CONSTRUCTION

In this section we present data resources for query content analysis and Arabic NLP to help deal with the complex nature of Arabic in order to build efficient Arabic information retrieval tools.

A. *Arabic, the Big Challenge*

Arabic is a highly inflected language with a rich and complex morphological system. Any given Arabic lemma usually has more than one word representation [5]. Arabic NLP faces major challenges that are not necessarily shared with many other languages, challenges such as complex linguistic structure, the specific features of its orthographic system, and processing colloquial Arabic [6]. This in turn adds complexity to retrieving information using Arabic language.

B. *Arabic Corpus Construction*

The development of NLP tools and methods needs the availability of extensive and reliable text corpus. Throughout the process of developing our Arabic NLP tools we employed a statistical/Corpus approach based on contemporary data we obtained from various sources (newspapers: Al-Sharq Al-Awsat , Al-Quds newspaper). The corpus of news articles had around 75 million words of written Arabic in 80,000 pages covering different topics. The Corpus construction was carried out by crawling Al-Sharq Al-Awsat newspaper website (http://www.aawsat.com/), and Al-Quds newspaper PDFs (http://www.alquds.com/pdf). A filtration method is used for the enhancement of the text by extracting numbers, punctuations, diacritics and Shadda (ّ). Data statistics on our corpus are shown in Table I.

C. *Construction of Arabic Stop Word List*

Stop words is a list of very common words which are filtered prior to/after processing of natural language data [7].

TABLE I. DATA STATISTICS

Description	Statistics
Processed Words	75,132,120
Arabic Words (no repeat)	962,879
Arabic Words (F > 1) [a]	519,827
Multi words expression (no repeat)	1,843,274
Triple words expression (no repeat)	1,414,010
Number of documents (PDFs , HTML)	Around 80,000
Average letters per word	5.4 letter
The most frequent word	1,203,663 (في)
Number of letters for the longest word	15 (الكهرومغناطيسية)

a. F = Frequency of appearance. Words are considered different if they differ in shape (no stemming or letters filtering is done).

We created an Arabic stop word list that consists of the Arabic prepositions, pronouns, interrogatives, particles, words with the highest frequencies from our text corpus database and words translated from English stop words list[8] using Google online translator. Also an open source Arabic stop word list was integrated later into our list [9].

The Arabic stop words list has 1065 words, which is a large number compared to English which has around 320 stop words [8]. That is because Arabic has much richer morphology than English, Arabic has two genders (feminine and masculine), and three numbers (singular, dual and plural) and sometimes pronouns and prepositions are joined together to form new words. So pronouns, prepositions and frequent words can have more than one form. For example the word (in- في) can have the following forms (وفيه، وفي ،فيه ، فيها ، فيهم ،فيهما ، وفيها ، وفينا ، وفيك ، فيك ، فينا، ففي، فيكم ،ففيها،فيهم، فينا، وفيهما،وفيك، فيها،ففيهما، ففيه،فيكم، وفيكم ، فيكن ، وفيكن، و فيهن، وفيهما، فيهم، ففيكن،.. الخ)

Each such Arabic word will translate into multiple English stop words: and in it, and in, in it…etc.

IV. ARABIC NATURAL LANGUAGE PROCESSING TOOLS

In this section we present some Arabic NLP tools that we built and tested toward the goal of improving information retrieval in Arabic. This is a partial collection of the tools we worked on and may be viewed as the infrastructure for other work.

A. *Arabic Language Detector*

This allows us to determine if the language of the web document is Arabic, and not any other languages that uses Arabic script, say Persian or Urdu. Such tool comes handy to crawlers in order to crawl and index the Arabic web contents only. The automatic language detector determines the language of the document or query by comparing the words in the document/query with the words in our (partially built) corpus and calculating the percentage of misspelled words. A derivative tool was a plug-in to restore Arabic text entered in Latin due to failure to switch keyboard entry language.

B. *Arabic Query Live Suggestion*

In order to make the search process interactive, we built query suggestion feature. When the user types in the search box, the system queries the suggestion tables to bring a list of possible completions/alternatives. Once the list has been retrieved, it is displayed in a pop-up box that appears under the search box, and allows the user to choose a suggested search term (check Fig 1). If the user continues to type, a possibly new set of suggestions may be displayed. This will limit the number of words users' type into a query, present similar queries and eliminate typos, and may introduce a learning component into the interaction, something that may speed up the search process.

C. *Stemming and Root Extraction*

Arabic is a highly inflected language which has as a rich and complex morphological system. Arabic words are usually

Figure 1. Live Suggestion Example

formed as a sequence of prefix, core, and suffix. Indexing the Web based on the roots, which are far more abstract than stems, will improve the retrieval effectiveness over stems and words. In this section we illustrate a new stemming and root extraction technique for Arabic words. Such a tool will help us build the expansion algorithm for Arabic quires. Arabic words are divided into three types: noun, verb and particle. Nouns and verbs are derived from a set of around 10,000 roots and they commonly consist of three or four, and rarely five letters [10]. Arabic words are formed by adding prefixes (consonants, vowels at the start), infixes (vowels) and suffixes (consonants and vowels at the end) to the root. So, finding the root basically means reversing the process of forming Arabic words by removing prefixes and suffixes, then predicting the root of the core word.

The form of an Arabic word is usually determined by its gender, number, grammatical case, whether it is definitive or not, and finally if there is a preposition attached to it.

Stemming is carried out in the following steps: We start by recursively removing prefixes and suffixes then attempting to find root for the stripped form. Our approach is to define seven level of processing (L3, L4, L5, L6, L7, L8, Ln) that the query may pass through during the process. The Number of characters in a word determines its starting level (for example, the word "ينتصر" will start at level five-L5). Words with less than three letters will not be processed and will be directly considered roots. Words with more than eight letters will start at level n. At each level of processing we considered the following hypotheses:

- Removing all possible prefixes and suffixes from a word will result in a word formed of three letters that we can consider as a root.
- Prefixes and suffixes are either one, two or three letters.
- More than one prefix or suffix can be attached to the word.
- Level four processing addressed infixes processing. In addition it takes into account one letter prefix or suffix. After that the output is sent as a three letters word to L3.
- Words are composed of: prefix(es), a stem, and suffix(es). [Prefix (0-6 letters)-stem (1-4 letters)-suffix (0-6 letters)].

At each level of processing, all the possible combinations of (prefix core suffix) are examined and for combinations where the core is a correct Arabic word, prefix and suffix are extracted and the cycle moves on until the word has four letters.

Then the word enters level four processing which checks it first for infix then for suffix or prefix (see Table II). After deleting any infix found or any one letter suffix or prefix, the word is sent to L3 which firstly checks if there is a vowel (ا، و، ي)in the word. If there is, the word is expanded to three shapes: one with "ا "as replacement of the word vowel, the other two shapes for "و " and "ي". The same occurs if the word has Hamza in it, words expanded to have all the possible shapes of Hamza. Then the word and its expansions are compared to list of 6,000 roots (Tim Buckwalter root list) [11], and the most similar root is considered as root for the word.

For example processing وسيأخذونهما is carried out in the following steps: LN → وسيأخذونهما then L7 → سيأخذون then L3 → أخذ

We omit the fine details here about how our stemmer works for space considerations.

Validation experiments have been carried out to evaluate the performance of our root extractor. We selected 500 words randomly and ran the root extractor on them. The result was that 49 out of 500 words failed the test, which means that the overall performance of our root extractor is around 90% accurate, however this is a preliminary result with words that will converge to 3-letter root only.

D. Query Expansion Tool

If an Arabic retrieval system restricts its search to the exact query without looking for the relevant words or derivatives, the results will be poor. To overcome this, expansion techniques are used in search engines, making use of the fact that in Arabic, many words can be derived from a single root. The Query Expansion builds expanded queries from roots: for example if we have the word يدرسون then expanding it will give words such as مدرس ، دراسة ، يدرسونهما ، يدرس and so on. What's common with these words is the root of the original word, درس. So in order to have an expansion tool to use with queries we first need to have tables that relate each word in Arabic with its root. To build such tables we used our corpus and Arabic root extractor system. Also stop words in a query should be removed by a stop word filter. The foreign names will be kept as they are, the root extractor will tag them as unprocessed words.

To overcome the problem of the representation of Arabic letters (usually resulting in common errors: hamza shapes, alef, shapes ..) we applied normalization rules when expanding the input query , that is to match between " ه " and " ة " in the end of the word. For example if the user query holds the word مدرسه then we should search for مدرسة and مدرسه plus expanded words related to both. Same said about " ا " ، " أ " ، " آ "، " إ "، "" in the first of the word and " ي " ، " ى " in the end of the word.

E. Query Correction and suggestion

The main function here is to correct user entered queries.

TABLE II. POSSIBLE PREFIXES, SUFFIXES AND INFIXES THAT MAY ATTACH TO ROOTS TO FORM WORDS

Prefix	Suffix	Infix
نست ، يست ، تست ، است ، سأ ، سن ، سي ، ست ، ن ، ي ، ف ، ت ، م ، ل ، ب ، أ ، و	كما ، هما ، ان ، ون ، ين ، ية ، هن ، هم ، ة ، ي ، وا ، اء ، ن ، ت	ا ، ي ، و ، ت

This has the flavour of spell checking techniques using dictionary lookup. Here, we first test the correctness of the query by looking for matching words (regardless of their order) in the dictionary. If there was a match, the query is considered correct; otherwise, the dictionary looks for a list of possible replacements. Such replacements might be based on the fact that in Arabic, one can find words with different spelling still pronounced in the very same way, and errors that occur as a result of similar pronunciation or spelling.

In order to build a spelling system for search query we need three types of corpora, single, double and triple expressions, why? Let's take the following example: the spelling of الوظن الغربي , the word الوظن is wrong and the word الغربي is correct , however if spelling each alone the word الوظن will be spelled either الوطن or maybe الوزن . That depends on the ranking system, but in the case of double expressions spelling, the system will look at the expression as one block and detect that the best solution is الوطن العربي neither الوطن الغربي nor الوزن الغربي. Here, the double expressions were useful; same is said about the triple expressions. The double and triple expressions were built from the original newspapers pages.

Our correction algorithm depends mainly on Levenshtein distance which works as a metric measurement that gives the number of steps (minimum) needed to convert string A to string B[12]. Another important component of the correction algorithm is the ranking system that takes different measurements in consideration while sorting possible correct outputs to misspelled input. The ranking system takes the following (weighted) parameters in consideration:

1) Shape Similarity: A function that measures the similarity in shape between two words. For example, if the input misspelled word is الوظن and the spelling algorithm gave out two possibilities: الوزن and الوطن then the shape similarity function will detect that the word الوطن looks more familiar in shape to the word الوظن since ط and ظ hold the same shape, and letter ز doesn't look the same.

The similarity function will split the input string into individual characters; each character is compared with predefined groups of letters with similar shape. Table III defines these groups.

If the character is in the Nth group then the group code is given to that character, till all characters are replaced by group codes. For example, the word درس has the word code: 551, and the word ذرس has the same code 551. The same applies to ذزس and ذزش, which results in 0 (no difference) shape differences between the above three words, however if we are comparing مدرسة with ورشة the function will give مدرسة C5518 and will give ورشة two codes either 0D518 or D5180 (the difference is in the empty letter position, which is used to make both words equal in length). Comparing C5518 with 0D518 will give 2 shape differences and comparing C5518 with D5180 will give 4 shape differences. In this case the function chooses the lowest difference code and the final output will be the 2 differences. As a percentage that indicates the relation between both inputs, the output will be:

ShapeSimilarity= numberOfRelatedLetters/lengthOfLargestInput (1)

The word shape similarity codes will be used to compare words in the spelling databases (sorted by appearance frequency) with the input "misspelled word" in order to get the best output word that is also the closest in shape (word with minimum difference and highest appearance frequency that match a misspelled input).

2) Location measurement: A function based on characters locations on the keyboard. For example, the letter ا will have the following group: ت ل ف غ ع ة ى which are the letters located around it in the keyboard, and so on for other letters. This measurement will give a numerical value that expresses the relation between two words based on their characters' locations on keyboard. In our work the location measurement function is limited to PC keyboards, though it can be extended to other layouts.

The function compares two words (misspelled and potential correction) to measure how much both are related due their letters locations on the keyboard. Groups of related-by-location letters are used in the detection, the function starts by taking word with smallest length to be the base, a letter in one word is compared with the same location letter in the other word: (assume the letters are X1 and Y1), if X1=Y1 then a counter *equalLetters* that gives the number of equal letters is incremented, If X1 and Y1 are related by location on the keyboard (related is measured by being neighbor letters) another counter *relatedLetters* is incremented that gives indication about related letters. This is done for all letters, and then the following measurement equation is applied:

LetterLocation= (equalLetters +R*relatedLetters) /length (longest Word) (2)

Where R value is considered to be 1, which means equal letters and related letters have the same weight. However for

TABLE III. SHAPE SIMILARITY GROUPS AND CODES

Group	Code	Letters		Group	Code	Letters
1	1	س ش ص ض		9	9	ب ت ث ن
2	2	ط ظ		10	A	ك ل
3	3	غ ع		11	B	ج ح خ
4	4	ف ق		12	C	م
5	5	ر ز د ذ		13	D	و ؤ
6	6	آ ا أ إ		14	E	ء
7	7	ء ئ		15	F	ئ ى ي
8	8	ة ه		16	0	No letter

future work and experiments R value can be changed to give the related letters more or less weight than the equal letters.

Considering مدرسة and كدؤسة the function will detect 3 equal letters and 2 related letters with measurement equation [3+P(2)]/5 = 1.

This means that both words are fully related when it comes to letters locations on keyboard.

3) Soundex Function: Originally, this is a "phonetic algorithm for indexing names by sound as pronounced in English"[13]. In our case in Arabic, the idea of the algorithm is to look for groups of words that have the same sound somehow and replace them by a certain code. And any two words that have the same code are considered a match in sound such as كلينتون، كلنتون.

The soundex function works much like the shape similarity function, it splits the input string into characters; each character is compared with predefined groups of letters each holding letters with same sound, then each letter is replaced with its group code. (Table IV defines the groups).

Consider the words كلينتون، كلنتون. The first will give the code: 3KM26M and the second will give: 3K6M26M , then the function removes letters that hold the code 6 (Arabic vowels) which gives 3KM2M for both words and thus the words will match and are considered related by sound.

However, if the codes for the compared inputs don't match, the soundex function will give the percentage of matched letters in both.

$$Soundex = (matchedLetters/\#ofLettersInLargestInput) \quad (3)$$

The equation for ranking a possible output word from the Levenshtein distance will be like this:

$$Rank\ (word) = A*Frequency + B*ShapeSimilarity + C*LetterLocation + D*Soudex) \quad (4)$$

Where A, B, C, D are percentages with summation of 100% (weights). Consider A = 0.5 and B = 0.20 and C= 0.25 and D = 0.05 in which case the equation will be:

$$Rank\ (word) = 0.5*Frequency + 0.2*ShapeSimilarity + 0.25*LetterLocation + 0.05*Soudex) \quad (5)$$

The chosen values for A, B, C and D are not necessarily the best. They are based on experimentation and thus need more testing to decide the best range (or values) for them. Tables V and VI show the result after testing some queries using the query correction

As mentioned earlier, the material reported here should be viewed as work in progress. Our query correction algorithm is being developed with extensive testing. Some of our early tests are based on auto generate errors of input queries and speed typing, then comparing correct inputs (they were manually checked for correctness) with correction algorithm outputs (the outputs generated due to error based inputs) to get pass/fail percentages.

TABLE IV. SOUNDEX FUNCTION GROUPS AND CODES

Group	Code	Letters		Group	Code	Letters
1	1	س ص ث		13	D	خ
2	2	ط ت		14	E	د
3	3	ك ق		15	F	ر
4	4	أ ئ ء		16	G	ب
5	5	ز ذ		17	H	ع
6	6	ا و ي		18	I	غ
7	7	ء ئ		19	J	ف
8	8	ة ه		20	K	ل
9	9	ظ ض		21	L	م
10	A	آ		22	M	ن
11	B	ج		23	N	ه
12	C	ح		24	O	إ

TABLE V. SINGLE WORD QUERY TESTING

#	Input	Output(s)
1	خامعة	جامعة ، خاضعة ، خامسة ، لامعة
2	المنتده	المنتدى ، المنتدب ، المنددة ، المتحدة
3	الخمهوزنه	الجمهورية
4	والقوانين	والقوانين ، والقاطنين
5	المستفيين	المستويين ، المستفتين ، المستفيدين
6	الضوفال	الصومال ، الأطفال ، الوفاء ، الوفاق
7	اليساسيين	السياسين ، السيادية ، الصيادين ، الميادين
8	ويتعاولنون	ويتعاونون

TABLE VI. DOUBLE AND TRIPLE EXPRESSIONS QUERY TESTING

#	Input	Output(s)
1	تفنته المغلوفات	تقنية المعلومات
2	الترق الاشظ	الشرق الأوسط ، الطرف الأخر
3	اللغفات الاجبية	اللغات الأجنبية
4	مخاملات خاتفيت	مكالمات هاتفية
5	خمياية اللكية التكجارية	حماية الملكية التجارية
6	رابطية الجامعت الاشلاميه	رابطة الجامعات الإسلامية
7	فن الوفواكه الخضار	من الفواكه والخضار
8	المشاركوت في المروتمر	المشاركون في المؤتمر

To perform such tests we selected 300 random queries divided into three groups of 100 queries each, from single words database, double expressions database and triple expression database. For each group, first a speed typing error based file was generated, which is a manual rewriting of each word in each group blindly (without looking at the keyboard) and the second is an auto generated error file, the file is created by replacing one letter in every word of the input file each time in a random position, three types of error files are generated: one error, two errors and three errors per word. Table VII gives an indication about the early tests and the pass percentages. The tests are made with A, B, C, D values (0.5, 0.20, 0.25, and 0.05), respectively.

Future work on spelling will include tests based on OCR output to test the efficiency of the algorithm in working with Arabic OCR and tests that measure the effect of each ranking variable alone. Also increasing words in the correction databases will be considered. We will also deal with ranking variables weights. Expanded tests that run over all possible values for weights of the ranking variables will be performed to decide which are the best values to use.

TABLE VII. EARLY TESTS ON QUERY CORRECTION ALGORITHM

#	Test Type	Query Type	Pass Percentage[a]
1	Speed Writing	Single	75%
		Double	85%
		Triple	84%
2	Auto generated errors		
	- One Error	Single	88%
	- One Error	Double	89%
	- Two Errors	Double	81%
	- Three Errors	Double	81%

a. These percentages are a subject of change by future tests and improvements.

TABLE VIII. EARLY TESTS ON QUERY CORRECTION ALGORITHM

Major Category	Subcategories
رياضة	كرة قدم ، كرة سلة ، تنس ، رالي وسباقات ، أولمبياد
هندسة	ميكانيكية ، أنظمة حاسوب و شبكات، كهربائية ، مدنية ومعمارية
علوم	رياضيات، كيمياء، أحياء، فيزياء
طب	صيدلة ، طب وصحة
فنون	شعر و أدب ، موسيقى و غناء، مسرح و سينما
ديانات	إسلام، مسيحية، ديانات أخرى
تاريخ	----------
سياسة	----------
إدارة أعمال و أموال	----------
موضة و مرأة	----------

V. AUTOMATIC ARABIC DOCUMENT CATEGORIZATION

The idea here is to map a document into one of a given set of categories based on text properties. The challenge is to correctly predict the category of the document even when the categories are related. We have done major experimentation on categorization algorithms that take into account the nature of Arabic and build on successes in other languages. Our experiments are based on statistical data, using sets of predefined words and their frequencies, sets that are defined as major categories and refined subcategories. These sets or words databases are obtained from different Arabic web based resources that are already categorized (such as sport, economics, medicine, political news websites) and from Arabic Wikipedia with manual defining, grouping and categorizing pages. Refer to Table VIII for major categories and subcategories that we are experimenting with.

Experiments that were done and others yet to be done vary from testing major categories and subcategories to testing the differences between related subcategories (such as the relation and differences between هندسة ميكانيكية، فيزياء، هندسة كهربائية or between أحياء، صيدلة، طب وصحة), also experiments vary between tests made using stemming, double and triple predefined and categorized expressions, window based categorizing (categorizing a limited number of words in a text and apply the results to the whole text) and other tests that our future work will explain in details. Our future work will explain how category based databases where built, frequencies of words in each, common words between related and nonrelated categories; we will describe each experiment that has been made and the results achieved. For space considerations; we are not able to give the details here.

VI. CONCLUSION

We presented the challenges that the Arabic language introduces for retrieving Web information and some Arabic NLP tools and methods that might help removing the barrier resulting from the sophisticated nature of Arabic.

Our work was aimed to develop techniques for returning good search results by helping search engines better understand users' queries and adding features to what currently exists in search engines. This paper reported on our work on designing text mining and query pre-processing tools that are able to efficiently process and search large quantities of Arabic web data. We employed a statistical/Corpus-based approach, and constructed databases that contain Arabic words from newspapers with their frequencies and used that as basis to create well structured search aid tools that are able to handle Arabic content and which are capable of being integrated into existing web search engines and document processing systems.

The reader may like to visit our website (*http://wojoodapi.appspot.com/*) which includes information about the tools we are working on (new tools also). One can also find some testing applications, worth mentioning here that we are working on a new version of the website that will show in detail our work and provide clear testing tools.

REFERENCES

[1] Search Engine Marketing and Internet Searching, Pandia, Search Engine News, "The Size of the World Wide Web", February 2007, http://www.pandia.com/sew/383-web- size.html.

[2] Thomas Boutell, "WWW FAQS: How many websites are there?" February 2007. [Online]. Available : http://www.boutell.com/newfaq/misc/sizeofweb.html.

[3] Grossan, "Search Engines, What they Are, How They Work, and Practical Suggestions for getting the Most out of them", [Online]. Available : http://www.webreference.com/content/search/, February 21, 1997.

[4] Arasu, Cho, Garcia-Molina Paepcke, Raghavan, "Searching the Web", *ACM Transactions on Internet Technology*, Vol. 1, No 1, pp. 2-43, 2001.

[5] Bies, Kulick, Maamouri, "Diacritization: A Challenge to Arabic Treebank Annotation and Parsing", University of Pennsylvania, USA. in proceedings of the Arabic NLP/MT Conference, The British Computer Society Natural Language Translation Specialist Group, 2006, pp.35-47.

[6] Adnan Yahya: "On the Complexity of the Initial Stages of Arabic Text Processing"; *First Great Lakes Computer Science conference*; Kalamazoo, Michigan, U.S.A.; 18--20 October 1989.

[7] wikipedia Stop words, Wikipedia Website.[Online]. Available: http://en.wikipedia.org/wiki/Stop_words/ ,[Nov, 20, 2009].

[8] Stop words, University of Glasgow, Department of Computing Science, information retrieval resources.[Online]. http://www.dcs.gla.ac.uk/idom/ir_resources/linguistic_utils/stop_words/, [Oct, 10, 2007].

[9] Wikipedia,Stop words, Arabic Stop Words Project.[Online].Available: http://sourceforge.net/projects/arabicstopwords/, [Nov,20,2009].

[10] Kareem Darwish, "Building a Shallow Arabic Morphological Analyzer in One Day", ECE Department, University of Maryland. [Online].Available: http://www.cs.umd.edu/Library/TRs/CS-TR-4326/CS-TR-4326.pdf

[11] Tim Buckwalter, "Arabic root list",1997.[Online].Available: http://www.angelfire.com/tx4/lisan/roots1.htm . [Mar,4.2011].

[12] Wikipedia, Levenshtein Distance, Wikipedia Website.[Online].Available: http://en.wikipedia.org/wiki/Levenshtein_distance/,[Mar, 5,2011].

[13] Wikipedia, Soundex, Wikipedia Website.[Online].Available: http://en.wikipedia.org/wiki/Soundex, ,[Mar,5,2011].