Proceedings of the 21st IASTED International Conference
APPLIED INFORMATICS
February 10-13, 2003, Innsbruck, Austria

AN ARABIC AUTO-INDEXING SYSTEM FOR INFORMATION RETRIVAL

Ramzi A. Haraty, Nashat Mansour, and Walid Daher
Lebanese American University
P.O. Box 13-5053
Beirut, Lebanon 1102 2801

ABSTRACT

This paper tackles the problem of auto-indexing Arabic documents. For that purpose, a four-layer model is proposed as a solution. The model's layers are interdependent meaning that the proposed solution works fine not only for Arabic but also for any other language. Obviously, that would require the Arabic grammar to be taken into consideration. In addition, this report introduces a new concept to calculate the weight of a term relevant to its container document. Traditionally, the weight of a term used to rely totally on the rate of repeat (or the count) of that term. The new innovation is to take into consideration the rate of "spreading" within the document.

KEY WORDS

Information retrieval, stem word extraction, term's weight, and spread factor.

1. INTRODUCTION

Manual indexing is considered to be a cumbersome task for all people who work in the domain of information retrieval. People who perform indexing in a newspaper, magazine, or any other information resource, are specialist people, very well trained, and have a solid linguistic background. A solid background means that these people should be talented in speaking and talking the language, have rich vocabulary, and most importantly they should be experts in matters that concern the grammar of the language. The people responsible for doing this job are called documenters.

Indexing is of two types: Thesaurus based indexing and Full-Text based indexing [3]. In Thesaurus based indexing, the documenter may choose words to represent a document that do not even exist in the document. However, the synonyms do exist. Full Text based indexing, on the other hand, is much easier in concept. It totally relies on terms, as well as phrases, within the document itself. Nothing is imported.

The problem of auto-indexing varies in difficulty between one language and another. Languages with sophisticated grammatical rules such as Arabic or Chinese languages make the process of auto-indexing quite difficult. The only solution is to implement an algorithm that covers most of the grammatical rules, since writing an algorithm that covers all rules is very difficult, if not impossible.

Whether Thesaurus-based indexing or Full-Text indexing is used, the output is the same: "A set of keywords". Indexes, when extracted from documents, are referred to as "Key Words". Key words, in turn, are used to build "Subject headings". Usually, subject headings are phrases composed of one or more key words. A single document may have as many subject headings as possible. The more subject headings a document is assigned, the more likely that a user might hit that document upon searching for a topic. Composing subject headings is what documenters actually do. There are certain rules that documenters follow in order to build subject headings. A subject heading is composed of the following fields:

- *Name*. For example, the name of person or organization the document is about
- *Position* – Social position. For example, "President of Lebanon" or "President".
- *Country/City/Town/…Place*. For example, "Beirut, Lebanon".
- *Activity*. For example, "Meeting with prime Minister".

An example of a subject heading is the following:
George Bush>President>USA>Meeting

Therefore, documenters have two major tasks: 1- reading whole document and picking up candidate key words. 2- Building subject headings. Building subject headings is beyond the scope of this paper. However, the idea is to find a solution for picking up the key words, and hand them to the documenter to build up the subject headings. Thereby, more than half of the difficult work shall be automated. The solution is a four-layer model that performs auto-indexing and extracts valid indexes.

The rest of the paper is organized as follows. Section2 presents the solution. Section three discusses stem word extraction. Section 4 overviews weight calculation. In section 5 the index selection is presented. Section 6 concludes the paper.

2. SOLUTION – A FOUR-LAYER MODEL

The model proposed in this paper consists of four layers as seen in the figure 1. Each layer is a module that is implemented alone, and is totally independent from the other modules. The layers do exchange information, however. The output of one layer is the input for the above layer. Notice in the figure below that the second layer,

where words are stemmed to their original form, is drawn alone. The figure illustrates that the module that is responsible for stemming the Arabic words can be easily plugged off the system and replaced by another module that stems words in any other language. Every thing else will work just fine.

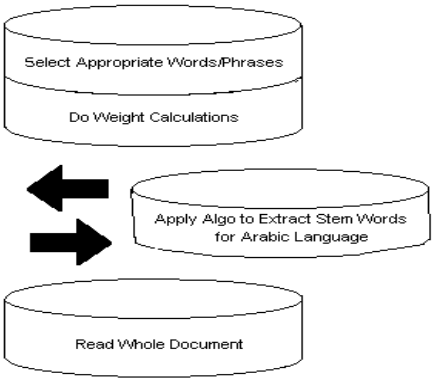

Figure 1. Four-layer Model

The first layer from the bottom is merely concerned about reading the document word by word and filling it into an array. Words that belong to the Stop-List terms are omitted. Stop list terms are words/phrases that occur within a document, yet they do not contribute to the meaning of the document whatsoever. Candidate words, on the other hand, shall undergo the layer above. At this layer, words are stemmed to their most probably three-letter form, and filled into another array. Thus, the output of this module is two sets of words (more technically, two arrays of strings): The first of which is called "*Words"* array. It is an array of records each record having four fields: 1-The word itself, 2- the stem word, 3- the frequency of the, and 4- the weight of the word (see figure 2 below). At this stage, the weight field is kept blank. It is updated in the third layer when all criteria for weight calculations are available. The second array is called "Stem Word" array. It is an array of records consisting of three fields. 1- Stem Word, 2- Count, and 3- Ideal distance (referred to as *ID*. For further details and clarification, see figure 3. The records structures for both arrays are illustrated in figure 2.

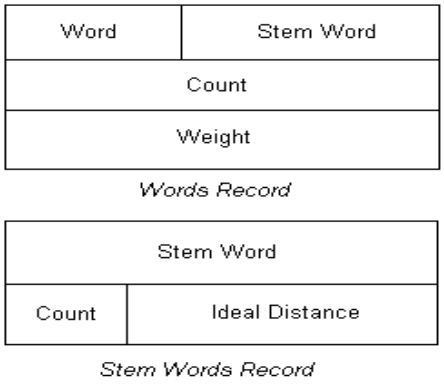

Figure 2. Record Structure

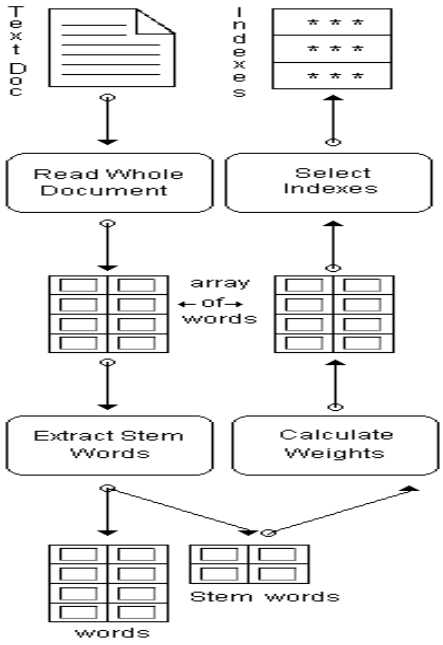

Figure 3. Workflow Diagram

Finally, in the top most fourth layer, the appropriate indexes are selected. The index selection technique may vary according to the overall purpose of the auto-indexing system.

3. STEM WORD EXTRACTION

Recall that the output of this module is two sets of words. The first of which is the list of the words that are candidate indexes, and the second set is the one that contains the corresponding stem words. Each stem word in the latter set may have one or more corresponding words in the former set.

Two tasks should be done before doing the stemming job. The first task is to decide whether a word is a stop-list term. This is done by simply having the algorithm check a word against all words that exist in the stop-list. If a word is a stop-list term, then it is ignored. Else, the word shall undergo the second task of the algorithm, which is to decide whether a word is a verb or a noun. This step is extremely important so that the algorithm would know what set of stemming rules shall be applied. It should be cleared out that the stemming rules applied to verbs are different than those applied to nouns. Now the question is: how does the algorithm decide whether a word is a noun or a verb? Basically, two things determine the type of a word. The first clue is the word preceding the word into consideration. This is the case especially if the preceding word is a stop list term. Some stop list terms precede nouns only; some others precede verbs only. The second clue is the rhythm of the word itself. If in case the algorithm was not able to decide the type of the word, then it returns the value "Unknown". In that case, the techniques applied to

nouns as well as verbs are applied separately. If either of both set of stemming techniques succeeds in stemming the word to its three-letter word form, then that word would be the stem of the initial word in consideration.

3.1 RHYMING ALGORITHM

The Rhyming Algorithm is a basic function for performing word stemming. It is not at the core of the word-stemming algorithm, but it is rather described as an essential utility for doing the stemming task.

When applying the Rhyming algorithm against a certain word, that word is compared to a special set of rhythms. The set of rhythms changes according to the module calling the rhyming algorithm. For example, the set of rhythms used to decide whether a noun is in its singular or plural form is different from the set used to determine the attached prefix/suffix pronouns.

One last point to mention before listing the algorithm itself is that all words are rhymed with the derivations of the word 'فعل'. The derivations of the verb 'فعل' have been used as a standard in all books that teach the Arabic grammar.

```
Boolean RhymeWords (Rhythm, Word) {
If Length (Rhythm) <> Length (Word) then
        Return False;  // words do not rhyme
Else {

        Len = Length (Rhyme)
        // Now Compare Rhythm and Word Letter By
        Letter
        i := 0;
        WordsRhyme := True ;
        While (i < Len-1) && (WordsRhyme) {
        //Ignore letters of word 'فعل' while
        Rhyming
                If Not (Rhythm(i) In ['ل', 'ع',
                'ف']) Then
                        WordsRhyme =
                        (Rhyme(i) == Word(i));
                i++;
                } // While
        }
        return WordsRhyme;
}
```

3.2 EXTRACTING STEM WORDS FROM VERBS

Recall that the whole idea behind knowing the type of a certain word is to know what stemming techniques ought to be used. This section lists and explains three stemming techniques that extract stem words from a verb. Remember that a successful algorithm that extracts stem words from verbs is the one that returns any verb to its original three-letter form.

3.2.1. CHECKING ATTACHED PREFIX/SUFFIX PRONOUNS

The first applied stemming technique is to check whether a word contains attached pronouns. Pronouns in Arabic language come in two forms: attached pronouns ('ضمائر متصلة') and discrete pronouns ('ضمائر منفصلة'). The discrete pronouns are considered as stop list terms, and thus the algorithm ignores them. The attached pronouns however, are part of the word itself. Hence, they should be spotted and identified by the algorithm in order to separate them from the verb. How does the algorithm realize that a verb has some attached pronouns? Simple… Attached pronouns appear either at the beginning of the word or at the end of the word or at both sides. The list of all attached pronouns is a finite and a defined set, and is listed in figures 4 and 5 [9]. The algorithm loops over the whole set of attached pronouns and performs pattern matching in order to check for the existence of any attached pronoun. In case it matches a pronoun, it removes it, and returns the verb barred from all suffix/prefix pronouns.

أفعل/سأفعل	الألف – للمتكلم	**يفعل/سيفعل**	الياء – للغائب المذكر
نفعل/سنفعل	النون – لجمع المتكلم	**تفعل/ستفعل**	التاء – للغائب المؤنث

Figure 4. Prefix Pronouns

فعلا	الألف - للمثنى الغائب	**فعلت/ك/ـه**	للمخاطب
فعلوا	الواو و الألف – لجمع المذكر الغائب	**فعلتما/كما/ـهما**	للمخاطب المثنى
فعلت	التاء – للمؤنث الغائب	**فعلتم/كم/ـهم**	للمخاطب جمع الذكر
فعلتا	التاء و الألف – للمؤنث المثنى الغائب	**فعلتن/كن/ـهن**	للمخاطب جمع المؤنث
فعلن	النون – نون النسوة		
فعلت	التاء – تاء مفرد المتكلم	**فعلنا**	النون و الألف – نون الجمع المتكلم

Figure 5. Suffix Pronouns

In addition to the above stated pronouns in the second table, things may even get more complicated when combinations of these pronouns occur together.

3.2.2 CHECKING VERB AGAINST COMMON FIVE VERBS

This section deals with what is known as the common five verbs in Arabic language. The common five verbs are known as 'الأفعال الخمسة'. These verbs come in a special form and have special properties: They always come in the present tense, and they always end with the letter 'ن'. If, however, these verbs are preceded with either 'أدوات نصب' or 'أدوات جزم' then the 'ن' letter must be removed [8]. The

five verbs are listed in figure 6 with and without the 'ن' letter at the end of the word.

الأفعال الخمسة- الصيغة الأولى	الأفعال الخمسة- الصيغة الثانية	الشرح
يفعلون	يفعلوا	جمع المذكر الغائب
تفعلون	تفعلوا	جمع المذكر المخاطب
يفعلان	يفعلا	المثنى الغائب
تفعلان	تفعلا	المثنى المخاطب
تفعلين	تفعلي	المؤنث المفرد المخاطب

Figure 6. Common five verbs (الأفعال الخمسة)

Unlike the attached pronouns, the algorithm does not perform pattern matching to detect whether a verb belongs to the five common verbs. Instead, it rhymes the verb against one of the ten mentioned rhythms above. If words rhyme, then the letters seen in red are the letters to be discarded, and the stem word would be the word composed of the black letters only.

3.2.3 CHECKING VERB AIGANST THE TEN VERB ADDITIONS

In Arabic language, every verb consists of only three letters. Verbs consisting of more than three letters are merely derivations of their original three-letter verb. The derivations of any verb occur in ten different formats. Three of these formats are obtained by adding a single letter on the original verb, five of them are obtained by adding two letters, and the other two formats are obtained by adding three letters. These ten formats, also named as derivations, are known in the Arabic grammar as ' الزيادات العشرة' [7]. The derivations, as well as an example of each of these derivations are presented in figure 7.

الزيادات	مثال	الأصل	الزيادات	مثال	الأصل
أفعل	أضرم	ضرم	إنفعل	إنهزم الأعداء	هزم
فعّل[1]	سرّع	سرع	إفتعل	إقترف خطأً	قرف
فاعل	قاتل	قتل	إفعلّ	إزهرّ الورد	زهر
تفعّل	تسبّب	سبب	إفعوعل	إغرورقت عيناه	غرق
تفاعل	تعاطف	عطف	إستفعل	إستخرج النفط	خرج

Figure 7. Ten Derivations (الزيادات العشرة)

Like the five common verbs, the algorithm detects one of the ten derivations by rhyming it with all the ten rhythms mentioned in the above table. If the algorithm detects that some verb is in the form of one of those derivations, it extracts the stem word by removing the letters colored in red (see the above table).

[1] The stressing character – known as 'شدة' – is considered a letter by itself in the Arabic language.

4.1 DEFINING TERMS

Define certain terms that are used in the formulas:

- *N = count of all terms in document*
- *m = count of a certain word*
- *sm = count of stem words for a certain word*
- *f = spread factor*

(Remember: the more a term is spread, the larger its factor becomes.)

Therefore, the weight *w* of a certain word becomes:

$$w = m \times sm \times f$$

The next step is to find a formula for that factor such that it increases as the term spreads all over the document, and decreases as the term concentrates in a specific section. Define certain terms for spread calculation:

- *d (Distance)*: A distance of a term is simply its position in the document [2]. In other words, it is the count of words preceding it. For example, the distance of the very first term in the document is one.
- *ad (Average distance)*: Is the average of all distances for a stem word.

$$ad = \left\lceil \frac{\sum_{i=1}^{sm} d_i}{sm} \right\rceil \quad \text{where } d_i \text{ is the distance of the } i^{th} \text{ term.}$$

- *Id (Ideal distance)*: is the ideal distance between every two occurrences for each stem word. The ideal distance of course should be equal between every two similar stem words. If the distance between every two stem words equals the ideal distance, this means that the term is perfectly spread all over the document. The ideal distance for a certain term is:

$$ID = \left\lceil \frac{N}{sm+1} \right\rceil$$

- *aid (average ideal distance)* : Is the average of all ideal distances for a stem word.

$$aid = \left\lceil \frac{\sum_{i=1}^{sm} i \times ID}{sm} \right\rceil$$

Notice however, that $\sum_{i=1}^{sm} i = \frac{sm \times (sm+1)}{2}$ and $ID = \left\lceil \frac{N}{sm+1} \right\rceil$

Thus, *aid* becomes $\left\lceil \frac{N}{2} \right\rceil$. Notice that *aid* is independent of *sm*. As a result of that, we can deduce that all stem words has one same average ideal distance, which is totally dependant on *N*, the number of terms in the document.

Unlike all previously defined terms, the *aid* term is an attribute of a document rather than the stem word since the distance, ideal distance, and the average distance vary for different stem words, whereas *aid* remains constant.

Finally, define the gap *g* to be the difference between *aid* and *ad.*

$$g = aid - ad$$

4.2 WEIGHT CALCULATION

Notice that as *g,* the gap, decreases, this indicates that the term is perfectly spread all over the document. Hence, this should affect positively the weight of that term. The converse is quite true. As the gap increases, this indicates that the term is concentrated at a certain part(s) of the document. This obviously means that this term does weakly reflect the content of the document. Thus, as *g* increases, the weight of the term should relatively decrease.

Going back to the formula:

$$w = m \times sm \times f$$

Define *f* to be a function of $g \approx f(g)$ such that:

(1) $\alpha < f(g) < \beta$; $\alpha, \beta \in \aleph$; $\alpha < \beta$; $\alpha \geq 1$; $\beta \leq N$

(2) $\lim_{g \to 0} f(g) = \beta$ *(The maximum value)*

(3) $\lim_{g \to \infty} f(g) \longrightarrow \alpha$ (The *minimum value)*

Assumption: *f(g)* may be defined as:

$$f(g) = \alpha + \frac{(\beta - \alpha)}{K^{g}}$$

Where $K \in \Re^{+}; K > 1$ [2]

Verification:

(1) $\lim_{g \to 0} f(g) = \alpha + \frac{(\beta - \alpha)}{K^{0}} = \alpha + \beta - \alpha = \beta$

(2) $\lim_{g \to \infty} f(g) = \alpha + 0 = \alpha$

Therefore, the formula is true. Notice in the graph shown in figure 8 that plots the values of *f(g)* as *g* varies.

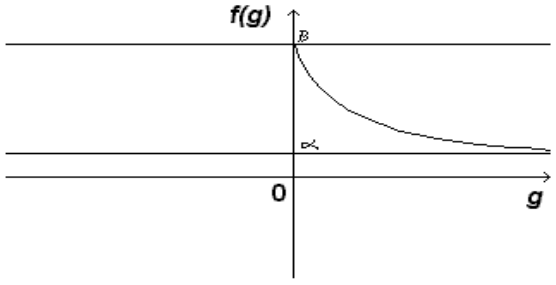

Figure 8. Graph 1 for spread factor f(g).

[2] Note that K could be any constant…the goal of the function is to get a factor which is a function of g, and between α and *β.*

However, the value of *g* could be positive or negative – as indicated before. Notice that this factor *f(g)* should have the same value had *g* been a positive value *x* or a negative value *-x*. This is because the average distance *ad* could have been *aid* - *x* or *aid* + *x*, and the gap would still have the value of *x*.

Thus the formula should be as follows:

$$f(g) = \alpha + \frac{(\beta - \alpha)}{K^{|g|}}$$

and the graph becomes as shown in figure 9:

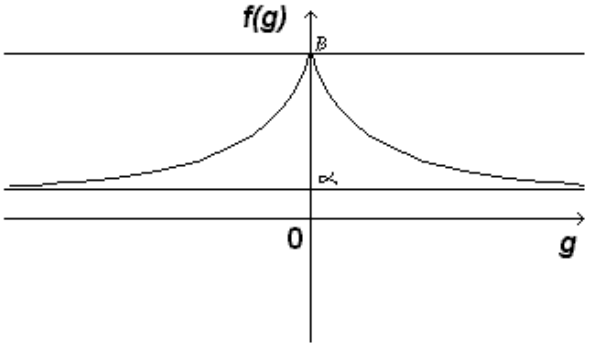

Figure 9. Graph 2 for spread factor f(g).

Thus the weight of the term would be:

$$w = m \times sm \times \left(\alpha + \frac{(\beta - \alpha)}{K^{|g|}} \right)$$

Practically, α and β are assigned their minimum and maximum values respectively. Additionally, K is assigned a value of 2. One thing was noticed regarding this constant that the larger K is chosen, the less significant value the weight formula will have. The final shape of the formula would be:

$$w = m \times sm \times \left(1 + \frac{N - 1}{2^{|g|}} \right)$$

5. INDEX SELECTION

Index selection is the ultimate level that a document undergoes upon auto-indexing. After all, this is what it's all about. All what ought to be done at this point is to select the indexes from the candidate terms. Actually what qualifies a term to be an index is basically its weight. The index selection mechanism varies according to the task that the overall auto-indexing system is assigned to do. For example, an auto-indexing system that is part of a general newspaper archiving system, may behave differently had the auto-indexing system been part of an Internet search

engine system. This difference in behavior between one auto-indexing system and another is embodied in the very last stage of the system, namely the "index selection" stage.

The index selection mechanism that is used by the auto-indexing system described in the introductory section of this paper was tailored for an archiving system of a news agency. In the aforementioned news agency, they receive online news, and publish it to newspapers as well as on the Internet at their site. Once the news is outdated, special people compile this news, and put it in archive. The archiving process that the documenters do is divided into three stages. During the first stage, a documenter reads the whole document, understands it, and selects the keywords that mostly mirror the document. In the second stage, the documenter merely matches some of the keywords that were retrieved with the corresponding keywords in thesaurus thus having a new enhanced set of keywords. Finally, the documenter composes the subject headings based on the previously mentioned rules (see the introduction), and feeds them to the Information Retrieval system in hand.

The index selection stage for this particular problem is basically to list all keywords for the documenter. This is what it's all about. The auto-indexing system reads the whole document, performs word stemming, calculates terms weights, and finally produce a list of keywords sorted descendingly by weight. In other words, the whole auto-indexing system is there to fulfill the first stage that a documenter has to do manually while preparing subject headings for a document.

6. CONCLUSION

The idea of auto-indexing varies in difficulty between one language and another since the module that roots words to their original stem depends absolutely on the grammar of the language in consideration. What was discussed in this paper is the following:

- A proposed model for auto-indexing that is composed of four interdependent layers. This model provides flexibility for the overall system since the system will not be bound to a specific language. In this report, the choice fell on the Arabic language. The overall system henceforth, performs auto-indexing on Arabic documents.

- A new criterion is suggested to be taken into consideration when calculating the weight, or relevance, of a certain term with respect to its container document. This new dimension is actually the level of spreading of a term in the document. This idea is based on the assumption that the more a word is spread in the document, the more likely it is to signify the document.

- Finally, and most importantly, the report contains new ideas in word extraction for Arabic words. This step is vital yet insignificant in languages that do not have sophisticated grammatical rules. As a matter of fact, the latter idea is the major contribution of this work to the field of auto-indexing Arabic documents. Rarely been the papers, or any other kind of work, that particularly tackled this problem. In this report, the stemming techniques discussed in section three are tailored to Arabic words, and conform to the Arabic grammatical rules.

REFERENCES

[1] A. Khurshid, L. Gillman, and L. Tostevin. Weirdness Indexing for Logical Document Extrapolation and Retrieval. *Proceedings of the Eighth Text Retrieval Conference (TREC-8)*. 2000.

[2] J. Labed, J. Deschnais, and A. Mili. Defining and Applying Measures of Distance between Specifications. *Department of Computer Science, Laval University, Quebec*, 1998.

[3] K. al Haqhaq. Head of Information Technology Department at KUNA – Kuwait News Agency – (*Personal Communication*). 2003.

[4] M. Siegler, R. Jin, and A. Hauptmann. Analysis of the Role of Term Frequency TF. *Proceedings of the Text Retrieval Conference (TREC-8)*. 1999.

[5] P. McNamee, and J. Mayfield. Indexing Using Both N-grams and Words. *Proceedings of the Seventh Text Retrieval Conference (TREC-7)*. 1998.

[6] V. Vijay and H. Shi. Evaluation of the 2-Poisson Model as a Basis for Using Term Frequency Data in Searching. *Proceedings of the Sixth Annual International ACM SIGIR Conference*, pp. 88-100. 1983.

[7] E. Deeb. The New Arabic Grammar Rules – Part Seven. Dar El-Kitab Al-Loubnany. Beirut. 1971.

[8] E. Deeb. The New Arabic Grammar Rules – Part Eight. Dar El-Kitab Al-Loubnany. Beirut. 1971.

[9] A. Koundary, F. Rajihy, and F. Chamry. The "Nahoo" Book – Part Seven. First Edition. Al-Risala Press. Kuwait. 1995-1996.