Available online at www.sciencedirect.com
ScienceDirect
Information Processing and Management 44 (2008) 1538–1545
INFORMATION PROCESSING & MANAGEMENT
www.elsevier.com/locate/infoproman

An auto-indexing method for Arabic text

Nashat Mansour *, Ramzi A. Haraty, Walid Daher, Manal Houri

Division of Computer Science and Mathematics, Lebanese American University, P.O. Box 13-5053, Chouran, Beirut 1102 3801, Lebanon

Received 4 July 2007; received in revised form 18 December 2007; accepted 29 December 2007
Available online 8 February 2008

Abstract

This work addresses the information retrieval problem of auto-indexing Arabic documents. Auto-indexing a text document refers to automatically extracting words that are suitable for building an index for the document. In this paper, we propose an auto-indexing method for Arabic text documents. This method is mainly based on morphological analysis and on a technique for assigning weights to words. The morphological analysis uses a number of grammatical rules to extract stem words that become candidate index words. The weight assignment technique computes weights for these words relative to the container document. The weight is based on how spread is the word in a document and not only on its rate of occurrence. The candidate index words are then sorted in descending order by weight so that information retrievers can select the more important index words. We empirically verify the usefulness of our method using several examples. For these examples, we obtained an average recall of 46% and an average precision of 64%.

Keywords: Arabic text; Document auto-indexing; Information retrieval; Stem words; Word spread

1. Introduction

Indexing text documents refers to selecting some words that represent the content of a document. The selected words are referred to as index words. Manual indexing of text documents is considered to be a cumbersome task in information retrieval. The people who perform indexing are usually well trained and have reasonable linguistic background. Manual indexing requires intensive human effort, since it requires people to read the whole document before selecting the candidate index words for that document.

There are two types of indexing: thesaurus based indexing and full-text based indexing. In thesaurus based indexing, the index words selected to represent a document might not exist in the document; but, their synonyms must exist. In this case, the index words are selected based on prior knowledge of what words might be searched for by users. In contrast, full-text based indexing is based on words found within the document itself. However, for both types of indexing, the output is a set of index (key) words from which an index for the

* Corresponding author. Tel.: +961 3 379647; fax: +961 1 867098.
E-mail addresses: nmansour@lau.edu.lb (N. Mansour), rharaty@lau.edu.lb (R.A. Haraty), daherwalid@hotmail.com (W. Daher), manoula78@hotmail.com (M. Houri).

0306-4573/$ - see front matter © 2008 Elsevier Ltd. All rights reserved.
doi:10.1016/j.ipm.2007.12.007

document can be constructed. These key words can also be used to define subject headings. Subject headings are phrases composed of more than one keyword. A single document may have many subject headings. The more accurate subject headings are, the more likely it will be for a user to hit that document upon searching for a topic in an information retrieval system.

Auto-indexing refers to automatic selection of key words in a text document. This problem varies in difficulty depending on the language used. Every language is characterized by its syntax, logical structure, and its domain (Harter, 1986). In particular, languages with sophisticated grammatical rules, such as Arabic, require sophisticated indexing methods.

A number of methods have been reported in the literature on related subjects. Classification algorithms for Arabic text have used the *N*-gram frequency statistic technique (Khreisat, 2006). An Arabic part-of-speech tagger that uses statistical and rule-based techniques has been proposed (Khoja, 2001). These parts-of-speech are further divided into nouns, verbs, and particles. The author uses a stemmer to remove all of a word's affixes to produce the stem. The stemmer faced problems when certain letters that appear to be affixes are part of the word, and when certain letters change to other letters when an affix is added. Gawrysiak, Gancarz, and Okoniewski (2002) describe the unigram and *N*-gram document representation techniques that are used frequently in text mining and discuss their shortcomings. They also present a technique that considers the word's position within a document. In Larkey and Connell (2001), the authors implement some standard approaches to handle co-occurrences. They also present several monolingual and cross-language runs and show that stemming improves their system. Term co-occurrence data are also used in Billhardt, Borrajo, and Maojo (2000) for document indexing. Rachidi et al. (2003) describe a multilingual Internet search engine that includes Arabic word stemming. Stemming is performed by removing postfix, infix, and prefix parts of words and by reducing singular, plural, and adjective forms into a canonical form. Larkey, Ballesteros, and Connell (2002) have developed light stemmers based on heuristics and a statistical stemmer based on co-occurrence for Arabic retrieval. Thus, very limited work has been reported on extracting indices for Arabic text.

In this paper, we present a full-text based auto-indexing method for Arabic text documents. Our auto-indexing method consists of three phases. The first phase consists of the processing steps: (a) apply rhyming step to classify all words; (b) determine and exclude stop-list words and phrases; (c) identify verbs and nouns. The second phase is concerned with extracting stem words from different types of verbs and nouns, which leads to determining candidate index words. The third phase is concerned with computing weights for the candidate index words relative to their document, which leads to selecting appropriate index words for the document. The usefulness of our method is demonstrated using empirical work involving several Arabic texts. For brevity and to serve general readers, we omit detailed language analysis.

The rest of the paper is organized as follows. Section 2 describes the preprocessing steps. Section 3 presents the word stemming algorithm. Section 4 describes how the weight of a word is calculated and discusses index word selection. Section 5 presents our experimental results. Section 6 contains the conclusions.

2. Phase 1 – preprocessing steps

Before extracting stem words, the first phase is a preparatory phase that involves preprocessing steps, which are described in the following subsections.

2.1. The rhyming step

The rhyming step preprocesses words by classifying them before stemming. Every word in the document is examined to decide whether it is a noun or a verb, whether a noun is in its singular form or plural form, whether a verb is in its past, present or future tense, and whether pronouns are attached to a word. In this step, every word is compared to an appropriate set of predefined rhythms. There are different sets of rhythms in Arabic. For example, the set of rhythms for used to decide whether a noun in singular or plural is different from the set for determining the attached pronoun. However, all words are rhymed with the derivations of the word 'Fa'ala' (i.e., did) and are marked with the relevant rhythm. The rhythms of the verb 'Fa'ala' are standard in the Arabic grammar.

2.2. Omitting stop-list terms and phrases

Stop-list terms are referred to as noise words. These words are those that do not contribute to the meaning of a sentence or a document, yet they help in forming a proper sentence (McNamee & Mayfield, 1998). Examples of such terms are 'never', 'it', 'the', 'that', 'where', 'numbers', etc. Stop-list terms are categorized according to their type by comparing them to predefined categories. Categorizing stop-list terms helps in determining the type of the word that follows, whether it is a noun or a verb.

Similarly to stop-list terms, stop-list phrases are sentences that occur within a document, yet they do not contribute to the meaning of the document. For example, a document may contain the phrase "Ladies and Gentlemen", yet it tells nothing about Ladies or Gentlemen. Stop-list phrases are detected by comparing the first word of the phrase with a certain set of words that hold the starting words of commonly used stop-list phrases. If a matching word is detected, then the rest of the phrase is compared with a set of stop-list phrases that begin with the same word.

2.3. Identifying verbs and nouns

Information retrieval systems usually perform natural language processing in order to auto-index a component. A component can be a document, an image, an audio information, etc.... Such processing depends on acquiring lexical, syntactic, and semantic information from that component and heavily involves the grammatical rules of the language considered (Franz & Roukos, 1998). Our method uses two clues in order to decide whether a word is a noun or a verb. The first clue is the word preceding the word in question. If the preceding word is a stop-list term, some stop-list terms precede nouns only whereas others precede verbs only. For example, verbs follow the 'lamm' (i.e., never) conjunction in Arabic whereas nouns follow prepositions. The second clue is the rhythm of the word itself. If, for example, a word rhymes with 'yaf'al' (i.e. does) or 'if'al' (i.e. do) then it is a verb. If it rhymes with the word 'fa'el' (i.e. doer) or 'maf'oul' (i.e. done), then it is most likely a noun. If then two clues fail, the attached pronouns are examined. Some pronouns are attached to verbs only, whereas others are attached to nouns only.

3. Phase 2 – stem word extraction

After traversing the whole document, identifying its words, omitting its stop-list terms and phrases, and identifying its nouns and verbs, a stemming algorithm is applied to each word, in the second phase. Stemming a word to its root term has been recognized as an important step for auto-indexing. Suppose that a certain noun comes once in the form of a singular noun and once in the form of a plural noun. Moreover, suppose that the same noun occurs once as an adjective, another time as a subject, or as an object. That is, the same word appears in different forms. If the auto-indexing algorithm does not perform word stemming, then it would treat each form of this noun as a different and independent term, which is incorrect. In this section, we present our stem extraction algorithm for both verbs and nouns.

3.1. Extracting stem words from verbs

Different verb types require different stemming procedures. The objective of any algorithm that extracts stem words from verbs is to return the original three-letter form of the verb. In this subsection, we describe how to stem different verb types. However, removing attached pronouns is a necessary first step.

3.1.1. Checking attached prefix/suffix pronouns

The first step in our stemming algorithm is to check whether a word contains attached pronouns. Pronouns in the Arabic language come in two forms: attached and discrete. Discrete pronouns are considered stop-list terms and are ignored. However the attached pronouns are part of the word itself. Hence, they should be identified in order to separate them from the verb. Attached pronouns come either at the beginning of the word or at the end of the word or on both sides. The lists of attached pronouns are finite and defined sets in Arabic, including prefix, suffix, and possible combinations. 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 stripped from all suffix/prefix pronouns.

3.1.2. Checking verbs against the 'five verbs'

In Arabic, there are five standard verbs known as the 'five verbs' (Deeb, 1971; Kindery, Rajihy, & Shimry, 1996). These verbs come in a special form and have special properties: they always come in the present tense and usually end with the letter 'N'. Further, these verbs have essential and non-essential letters. An example is 'yaf'aloun' (i.e., they do), where 'ya' and 'oun' are non-essential. 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 five rhythms. If words rhyme, then the non-essential letters are discarded, and the stem word would be the word composed of the remaining letters only.

3.1.3. Checking verbs against the ''ten-verb-additions"

In the 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 10 different formats. Three of these formats are obtained by adding a single letter to the original verb, five of them are obtained by adding two letters, and the other two formats are obtained by adding three letters. These 10 formats, also referred to as derivations, are known in the Arabic grammar as the 'ten-verb-additions' (Deeb, 1971; Kindery et al., 1996) and also have essential and non-essential letters like the 'five verbs'. Similarly to the 'five verbs' case, the algorithm detects one of the 10 derivations by rhyming it with the 10 rhythms. If the algorithm detects that a verb is in the form of one of those derivations, it extracts the stem word by removing the non-essential letters.

3.2. Extracting stem words from nouns

Extracting a stem word from a noun has some similar steps to stemming a verb, but is more complicated. The difficulty of stemming a noun is a result of many factors. One reason is that a noun may appear in the singular, double, or plural form. Also, each of these three formats differs when the noun addresses a male or a female. In addition, there are many exceptions for the double and plural formats. Furthermore, there may be plenty of derivations for a noun that have no specific format. The process of extracting a stem word from a noun is not described in this paper for brevity; details can be found in (Daher, 2002). However, the steps in the algorithm are: check attached prefix and suffix pronouns; check the noun against the 'five nouns'; restore a noun to its singular form; compare the noun to the 'common derivations': M-derivations (when first character is 'M'), T-derivations (when first character is 'T'), and miscellaneous derivations (when first character is not 'M' or 'T') (Deeb, 1971).

4. Phase 3 – weight assignment and index selection

4.1. Weight calculation

In the third phase of our method, we assign weights to the stemmed words relative to their document. The weight of a word depends on three factors, which represent the significance of a word in its container document. The first factor is obviously the frequency of occurrence of that word in its container document (Siegler, Jin, & Hauptmann, 1999). The second factor is the count of the stem words for that word. We propose adding a third factor, which is the spread of that word over the document. This suggestion is based on the conjecture that if a certain word is concentrated at a specific part of a document, then it is less likely that this word reflects its document had it been more spread in that document. This factor increases as the term spreads uniformly among all parts of the document. Likewise, it decreases as the term concentrates in a certain part of the document.

Consider the following terms that are used in the formulas for weight calculation: N is the number of words in a document, m is the count of occurrence of a certain word in a document, sm is the count of stem words for this word in a document, and f is the factor that indicates how much a word is spread within a document. The

weight w assigned to a word with respect to its container document indicates its significance to be selected as an index word. It is proportional to m, sm, and f and can be defined as:

$$w = m \times \mathrm{sm} \times f \tag{1}$$

The count of a word as well as of its stem words can be easily found by simply counting the occurrence of each in the document. Next, we develop a formula for the spread factor such that it increases as the word spreads over the document, and decreases as the term concentrates in a specific section.

Consider the following terms for spread calculation:

- d is the distance of a word. It represents the word's position in the document. In other words, it is the count of words that precede it plus one. For example, the distance of the very first term in the document is one. Likewise, the distance of the very last term is N. Also, d_i is the distance of the ith word.
- ad is the average of all distances for a stem word, defined as

$$\mathrm{ad} = \left[\frac{\sum_{i=1}^{\mathrm{sm}} d_i}{\mathrm{sm}}\right] \tag{2}$$

- id is the ideal distance between every two occurrences for each stem word. Obviously, the ideal distance should be equal between every two similar consecutive stem words. If the distance between every consecutive pair of stem words equals the ideal distance, this means that the word is perfectly spread over the document. The ideal distance for a specific word is given by

$$\mathrm{id} = \left[\frac{N}{\mathrm{sm}+1}\right] \tag{3}$$

- rid is a reference ideal distance for all stem words. rid is required to be the same for all stem words so that the actual distances of stem words can be measured with respect to this reference. Hence, rid shall be an attribute of the document rather than of specific stem words. Based in these considerations, several document-related choices are plausible. We choose rid $= (N/2)$ to be our reference ideal distance.
- g is the gap that is difference between rid and ad, i.e., $g = \mathrm{rid} - \mathrm{ad}$. Note that a decrease in g indicates that the word is perfectly spread over the document. Hence, this should positively affect the weight of that word. The converse is quite true. An increase in g indicates that the word is concentrated in a specific part(s) of the document: perhaps only in the first paragraph, the last paragraph, or even in one sentence. This obviously means that this word reflects the content of the document in weak way. Thus, as g increases, the weight of the word should decrease.

Performing weight calculation requires the use of f to be a function of g, $f(g)$, such that:

$$\left.\begin{array}{lll} \alpha < f(g) < \beta;\ \alpha,\beta \in \aleph; & \alpha \geqslant 1; \beta \leqslant N, & \\ \lim\limits_{g\to 0} f(g) = \beta & \text{(the maximum value)}, & \text{and} \\ \lim\limits_{g\to \infty} f(g) \to \alpha & \text{(the minimum value)} & \end{array}\right\} \tag{4}$$

Based on the above-mentioned considerations, $f(g)$ can be defined as

$$f(g) = \alpha + \frac{(\beta - \alpha)}{K^g} \tag{5}$$

where K is a constant such that $K \in \Re^+; K > 1$. Eq. (5) shows that $f(g)$ is a function of g that satisfies conditions (4) and its value lies between α and β.

4.2. Index selection

After assigning weights to candidate index words, what qualifies a word to be an index is its weight and not merely its frequency of occurrence. The index selection mechanism varies according to the task that the overall

auto-indexing system is supposed to do. For example, an auto-indexing system that is part of a general newspaper archiving system may have different requirements from an auto-indexing system that is part of an Internet search engine system.

One index selection technique is that used by some search engines, such as Internet search engines. In such information retrieval systems, all words extracted from the document are selected as index words. Actually, such systems work as follows: the document in consideration is assigned an identifier (ID), which is a unique value, and the latter is stored in the system's database along with the document's name, and its physical path. Subsequently, the system assigns every new word a unique ID, and inserts it into its database as well. However, all extracted words, whether new or already existing in the database, will be assigned an entry whereby the word ID is stored with the document ID along with the word's weight. In other words, the system will store the corresponding weight for each word where it occurs in each document separately (Convey, 1992).

The index selection technique we used is the one that can be used to create an index for a book. The index of any book is composed of key words that are alphabetically sorted and listed together with the page numbers where they occur within the book. Once the words within a document are stemmed, and the weights are calculated according to Eq. (1), the user may set a threshold for the weight to select the words for the index. That is, only the words whose weights are above the threshold will be retrieved and included in the index.

5. Experimental results

In this section, we describe and discuss our experimental results that show the usefulness of our auto-indexing approach. We use 24 arbitrarily selected general-purpose texts with various lengths. The total number of words in a text is denoted as N. For each text, we find the most useful index keywords by manual inspection, which is done by a domain expert; their number is denoted as I. Then, our method (described in Sections 2–4) is applied to retrieve significant index words from the text. Our method retrieves RR (retrieved) relevant index words and RI (retrieved) irrelevant index words. Hence, it will miss NRR (not retrieved) relevant words. That is, $\mathrm{NRR} = I - \mathrm{RR}$.

Table 1
Experimental results for 24 texts

Text no.	N	I	RR	RI	NRR	Recall	Precision
1	467	47	23	20	24	0.49	0.53
2	1044	95	47	17	48	0.49	0.73
3	789	43	21	10	22	0.49	0.68
4	532	56	27	16	29	0.48	0.63
5	532	58	31	20	27	0.53	0.61
6	546	51	22	16	29	0.43	0.58
7	587	60	24	19	36	0.40	0.56
8	887	97	34	16	63	0.35	0.68
9	1039	53	25	11	28	0.47	0.69
10	846	89	39	19	50	0.44	0.67
11	1023	86	34	23	52	0.40	0.60
12	714	73	43	13	30	0.59	0.77
13	767	89	44	20	45	0.49	0.69
14	513	47	24	12	23	0.51	0.67
15	413	40	21	14	19	0.53	0.60
16	424	46	25	18	21	0.54	0.58
17	463	41	19	10	22	0.46	0.66
18	577	53	24	10	29	0.45	0.71
19	559	56	23	18	33	0.41	0.56
20	496	40	21	13	19	0.53	0.62
21	451	62	29	16	33	0.47	0.64
22	599	60	19	14	41	0.32	0.58
23	633	60	26	19	34	0.43	0.58
24	720	70	26	22	44	0.37	0.54
Average						0.46	0.64

Table 1 shows our results for the 24 texts. To evaluate the results, we include the two usual metrics: precision and recall. Precision measures the percentage of relevant results; i.e., how well the retrieval algorithm avoids returning results that are not relevant. That is, precision = RR/(RR + RI). Recall measures the completeness of retrieval of relevant indices. That is Recall = RR/(RR + NRR) = RR/I.

The results given in Table 1 show that on average our method retrieves 64% of relevant index words and retrieves 46% of the manually determined index words (i.e. correct index words). Further, the precision and recall values for the individual texts do not vary much from the average. These are promising results for a challenging language like the Arabic language. That is, our method provides reasonable help to users who wish to construct indexes for Arabic documents. Actually, the recall results are better than what they appear to be since in all 24 cases the number of retrieved words (=RR + RI) is less than I, with respect to which the recall values are computed. Although our choice of the number of retrieved words leads to unfair comparison, it is kept in order to emphasize the automatic nature of our method.

We have observed that improving the results of our approach depends on addressing more challenges of the Arabic language. For example, some words have the same stem word, such as 'mal'ab' (i.e., playground) and 'yatala'ab' (i.e., twist the rules). But, these words have completely different meanings. Since our approach measures the distribution of the stem word and not of the word itself, it fails to detect such semantically different index words. Further, the use of 'tashkeel' (i.e., special symbols placed above and below letters) in Arabic words presents a unique challenge, where two words can be spelled the same way but they mean different things. An example is 'ilm' (i.e., science) and 'alam' (i.e., flag).

6. Conclusion

We have presented an information retrieval method that generates index words for Arabic text documents. The main part of this method consists of the word stemming algorithm (in the second phase) that employs Arabic grammatical rules. Another part (in the third phase) of this method introduces a spread factor that is computed and used for assigning weights to the words from which the final index words are retrieved. The experimental results carried out for a number of texts have demonstrated the advantage of our auto-indexing method.

Further research should enhance the stem word extraction algorithm. The Arabic language is a rich language in terms of its grammatical rules and the many exceptions that exist for some of these rules. Incorporating more of these rules will improve the recall and precision values of the indexing method. Also, the challenges mentioned in Section 5 can be the subject of further work.

Acknowledgement

The authors would like to thank Ms. Nebelah Haraty for her editing help.

Appendix. Implementing the weight calculation formula

The 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 rid $- x$ or rid $+ x$, and the gap would still have the value of x. Thus, Eq. (5) becomes:

$$f(g) = \alpha + \frac{(\beta - \alpha)}{K^{|g|}} \tag{6}$$

In implementing Eq. (6), α and β have been assigned the values 1 and N, respectively. In other words, α and β are assigned their minimum and maximum values, respectively. Also, K is assigned a value of 2. We have observed that the larger the K value is, the less significant the weight will become. The final form of Eq. (1) becomes

$$w = m \times \mathrm{sm} \times \left(1 + \frac{N-1}{2^{|\log g|}}\right) \tag{7}$$

References

Billhardt, H., Borrajo, D., & Maojo, V. (2000). Using term co-occurrence data for document indexing and retrieval. In *Proceedings of BCSIRSG 22nd annual colloquium on information retrieval research* (pp. 105–117).

Convey, J. (1992). *Online information retrieval: An introductory manual to principles and practice*. London: Library Association Publishing.

Daher, W. (2002). *An Arabic auto-indexing system*. M.S. Project Report, Lebanese American University.

Deeb, E. (1971). *New Arabic grammar*. Beirut: Lebanese Book Publishing.

Franz, M., & Roukos, S. (1998). Auto-indexing for broadcast news. In *Proceedings of 7th text retrieval conference* (pp. 115–120).

Gawrysiak, P., Gancarz, L., & Okoniewski, M. (2002). Recording word position information for improved document categorization. In *Proceedings of the PAKDD text mining workshop*.

Harter, S. (1986). *Online information retrieval: Concepts, principles, and techniques*. Orlando, FL: Academic Press.

Khoja, S. (2001). APT: Arabic part-of-speech tagger. In *Proceedings of student workshop at the second meeting of the North American chapter of the association for computational linguistics*.

Khreisat, L. (2006). Arabic text classification using *N*-gram frequency statistics: A comparative study. In *Proceedings of the 2006 international conference on data mining* (pp. 78–82).

Kindery, A., Rajihy, F., & Shimry, F. (1996). *Arabic grammar book*. Kuwait: Rissala Publishing.

Larkey, L., & Connell, M. (2001). Arabic information retrieval at UMASS. In *Proceedings of the 10th text retrieval conference* (pp. 562–570).

Larkey, L., Ballesteros, L., & Connell, M. E. (2002). Arabic information retrieval: Improving stemming for Arabic information retrieval: Light stemming and co-occurrence analysis. In *Proceedings of the 25th annual international ACM SIGIR conference on research and development in information retrieval*.

McNamee, P., & Mayfield, J. (1998). Indexing using both *N*-grams and words. In *Proceedings of 7th text retrieval conference* (pp. 419–424).

Rachidi, T., Iraqi, O., Bouzoubaa, M., Ben Al Khattab, A., El Kourdi, M., Zahi, et al. (2003). Barq: Distributed multilingual internet search engine with focus on Arabic language. In *Proceedings of the IEEE conference on systems, man and cybernetics*.

Siegler, M., Jin, R., & Hauptmann, A. (1999). Analysis of the role of term frequency. In *Proceedings of 8th text retrieval conference* (pp. 331–334).