
Adapting Morphology for Arabic Information Retrieval*

Kareem Darwish¹ and Douglas W. Oard²

¹ *IBM Technology Development Center, P.O. Box 166, El-Ahram, Giza, Egypt*
kareem@darwish.org

² *College of Information Studies & UMIACS, University of Maryland, College Park, MD 20742*
oard@glue.umd.edu

Abstract: This chapter presents an adaptation of existing techniques in Arabic morphology by leveraging corpus statistics to make them suitable for Information Retrieval (IR). The adaptation resulted in the development of Sebawai, a shallow Arabic morphological analyzer, and Al-Stem, an Arabic light stemmer. Both were used to produce Arabic index terms for Arabic experimentation. Sebawai is concerned with generating possible roots and stems of a given Arabic word along with probability estimates of deriving the word from each of the possible roots. The probability estimates were used as a guide to determine which prefixes and suffixes should be used to build the light stemmer Al-Stem. The use of the Sebawai generated roots and stems as index terms along with the stems from Al-Stem are evaluated in an information retrieval application and the results are compared

13.1 Introduction

Due to the morphological complexity of the Arabic language, Arabic morphology has become an integral part of many Arabic Information Retrieval (IR) and other natural language processing applications. Arabic words are divided into three types: noun, verb, and particle (Abdul-Al-Aal, 1987). Nouns and verbs are derived from a closed set of around 10,000 roots (Ibn Manzour, 2006). The roots are commonly three or four letters and are rarely five letters. Arabic nouns and verbs

* All the experiments for this work were performed while the first author was at the University of Maryland, College Park.

are derived from roots by applying templates to the roots to generate stems and then introducing prefixes and suffixes. Table 13.1 shows some templates for 3-letter roots. Tables 13.2 and 13.3 show some of the possible prefixes and suffixes and their corresponding meaning. The number of unique Arabic words (or surface forms) is estimated to be 6×10^{10} words (Ahmed, 2000). Table 13.4 shows some of the words that may be generated from the root *ktb* – كُتِبَ.

Further, a word may be derived from several different roots. For example the word *AymAn* – ايمان can be derived from five different roots. Table 13.5 shows possible roots for the word *AymAn* – ايمان and the meaning of the word based on each. For the purposes of this chapter, a word is any Arabic surface form, a stem is a word without any prefixes or suffixes, and a root is a linguistic unit of meaning, which has no prefix, suffix, or infix. However, often irregular roots, which contain double or weak letters, lead to stems and words that have letters from the root that are deleted or replaced.

Table 13.1. Some templates to generate stems from roots with examples from the root (*ktb* – كُتِبَ)

Template	Stem	Meaning
CCC – فعل	<i>ktb</i> – كُتِبَ	books, wrote, etc.
mCCwC – مفعول	<i>mktwb</i> – مَكْتُوب	something written
CCAC – فعال	<i>ktAb</i> – كُتَاب	book
CCACyC – فعاعيل	<i>ktAtyb</i> – كُتَاتِيْب	Qur'an school
CACC – كاتب	<i>kAtb</i> – كَاتِب	writer
CcwC – فعول	<i>ktwb</i> – كُتُوب	skilled writer

Table 13.2. Some example prefixes and their meanings

Prefix	w – و	K – ك	f – ف	l – ل	Al – ال	wAl – وال
Meaning	and	like	Then	to	the	and the

Table 13.3. Some example suffixes and their meanings

Prefix	h – ه	K – ك	hm – هم	km – كم	hA – ها	y – ي
Meaning	his	your (sg.)	Their	your (pl.)	her, its	my

Table 13.4. Some words that can be derived from the root ktb – كتب

Prefix	ktb – كتاب	wktAbh – وكتابه	yktb – يكتب	ktAbhm – كتابهم	mktbh – مكتبة	AlkAtb – الكاتب
Meaning	book	and his book	he writes	their book	library	the writer

Table 13.5. Possible roots for the word AymAn – ايمان along with meaning

Root	Meaning
Amn – أمن	peace or faith
Aym – أيم	two poor people
mAn – مأن	will he give support
ymn – يمن	Covenants
ymA – يماً	will they (fm.) point to

For Arabic IR, several early studies suggested that indexing Arabic text using roots significantly increases retrieval effectiveness over the use of words or stems (Abu-Salem et al., 1999; Al-Kharashi & Evens, 1994; Hmeidi et al., 1997). However, the studies used small test collections of only hundreds of documents and the morphology in many of the studies was done manually. Performing morphological analysis for Arabic IR using existing Arabic morphological analyzers, most of which use finite state transducers (Antworth, 1990; Kiraz, 1998; Koskeniemi, 1983), is problematic for two reasons. First, they were designed to produce as many analyses as possible without indicating which analysis is most likely. This property of the analyzers complicates retrieval, because it introduces ambiguity in the indexing phase as well as the search phase of retrieval. Second, the use of finite state transducers inherently limits coverage, which is the number of words that the analyzer can analyze, to the cases programmed into the transducers. A later study by Aljlal et al. (2001) on a large Arabic collection of 383,872 documents suggested that lightly stemmed words, where only common prefixes and suffixes are stripped from words, were perhaps better index terms for Arabic. Aljlal et al. did not report the list of prefixes and suffixes used in the light stemmer. If indeed lightly stemmed words work well, then determining the list of prefixes and suffixes to be removed by the stemmer is desirable. This chapter will focus on two aspects of Arabic morphology. The first is adapting existing Arabic morphological analysis using corpus statistics to attempt to produce the most likely analysis of a word and to improve coverage. The resulting analyzer is called Sebawai. The second is to construct a set of common prefixes and suffixes suitable for light stemming. The resulting light stemmer is called Al-Stem.

Section 13.2 will provide some background on Arabic Morphology and Arabic IR. Section 13.3 will provide a system description of Sebawai along with an

evaluation of its correctness and coverage. Section 13.4 describes the development of Al-Stem. Section 13.5 compares Sebawai and Al-Stem in the context of an IR application. Section 13.6 addresses some of the shortcomings of Sebawai and concludes the chapter.

13.2 Background

13.2.1 Arabic Morphology

Significant work has been done in the area of Arabic morphological analysis. Some of the approaches include:

1. The Symbolic Approach: In this approach, morphotactic (rules governing the combination of morphemes, which are meaning bearing units in the language) and orthographic (spelling rules) rules are programmed into a Finite State Transducer (FST). Koskenniemi proposed a two-level system for language morphology, which led to Antworth's two-level morphology system PC-KIMMO (Antworth, 1990; Koskenniemi, 1983). Later, Beesley and Buckwalter developed an Arabic morphology system, ALPNET, which uses a slightly enhanced implementation of PC-KIMMO (Beesley et al., 1989; Kiraz, 1998). However, this approach was criticized by Ahmed (2000) for requiring excessive manual processing to state rules in an FST and for the ability to only analyze words that appear in Arabic dictionaries. Kiraz (1998) summarized many variations of the FST approach.
2. Unsupervised Machine Learning Approach: Goldsmith (2000) developed an unsupervised learning automatic morphology tool called AutoMorphology. This system is advantageous because it could automatically learn the most common prefixes and suffixes from just a word-list. However, such a system would not be able to detect infix and uncommon prefixes and suffixes.
3. Statistical Rule-Based Approach: This approach uses rules in conjunction with statistics. This approach employs a list of prefixes, a list of suffixes, and templates to extract a stem from a word and a root from a stem. Possible prefix-suffix-template combinations are constructed for a word. Hand-crafted rules are used to eliminate impossible combinations and the remaining combinations are then statistically ranked. RDI's system called MORPHO3 utilizes such a model (Ahmed, 2000). Such an approach achieves broad morphological coverage of the Arabic language.
4. Light Stemming Based Approach: In this approach, leading and trailing letters in a word are removed if they match entries in lists of common prefixes and suffixes respectively. The advantage of this approach is that it requires no morphological processing and is hence very efficient. However, incorrect prefixes and suffixes are routinely removed. This approach was used to develop

Arabic stemmers by Aljlayl et al. (2001), Darwish & Oard (2002a), and Larkey, Ballesteros & Connell (2002).

13.2.2 Arabic Information Retrieval

Most early studies of character-coded Arabic text retrieval relied on relatively small test collections (Abu-Salem et al., 1999; Al-Kharashi & Evens, 1994; Darwish & Oard, 2002a; Darwish & Oard, 2002b). The early studies suggested that roots, followed by stems, were the best index terms for Arabic text. Recent studies are based on a single large collection (from TREC-2001/2002), (Darwish & Oard, 2002b; Gey & Oard, 2001) and suggest that perhaps light stemming and character n-grams are the best index terms. The studies examined indexing using words, word clusters (Xu et al., 2001), terms obtained through morphological analysis (e.g., stems and roots (Al-Kharashi & Evens, 1994; Aljlayl et al., 2001)), light stemming, and character n-grams of various lengths (Darwish & Oard, 2002a; Mayfield et al., 2001). The effects of normalizing alternative characters, removal of diacritics and stop-word removal have also been explored (Chen & Gey, 2001; Mayfield et al., 2001; Xu et al., 2001).

13.3 System Description

This section describes the development of an Arabic morphological analyzer called Sebawai, which adapts a commercial Arabic morphological analyzer called ALPNET (Beesley, 1996; Beesley et al., 1989) to find the most likely analysis and to improve coverage. ALPNET is based on a finite state transducer that uses a set of 4,500 roots. Sebawai uses corpus based statistics to estimate the occurrence probabilities of templates, prefixes, and suffixes. Sebawai trains on a list of word-root pairs to:

1. Derive templates that produce stems from roots,
2. Construct a list of prefixes and suffixes, and
3. Estimate the occurrence probabilities of templates, stems, and roots.

The words in the word-root pairs were extracted from an Arabic corpus.

13.3.1 Acquiring Word-Root Pairs

The list of word-root pairs may be constructed either manually, using a dictionary, or by using a preexisting morphological analyzer such as ALPNET (Ahmed, 2000; Beesley, 1996; Beesley et al., 1989) as follows:

1. Manual construction of word-root pair list: Building the list of several thousand pairs manually is time consuming, but feasible. Assuming that a person who

knows Arabic can generate a root for a word every 5 seconds, the manual process would require about 14 hours of work to produce 10,000 word-root pairs.

2. Automatic construction of a list using dictionary parsing: Extracting word-root pairs from an electronic dictionary is feasible. Since Arabic words are looked up in a dictionary using their root form, an electronic dictionary can be parsed to generate the desired list. Figure 13.1 shows an example of a dictionary



Fig. 13.1. An example of a dictionary entry where the root is inside a rectangle and words derived from the root are circled

entry for a root and words in the entry that are derived from the root. However, some care should be given to throw away dictionary examples and words unrelated to the root. Further, the distribution of words extracted from a dictionary might not be representative of the distribution of words in a text corpus.

3. Automatic construction using a pre-existing morphological analyzer: This process is simple, but requires the availability of an analyzer and a corpus. It has the advantage of producing analyses for large numbers of words extracted from a corpus.

For the purposes of this research, the third method was used to construct the list of word-root pairs. Two lists of Arabic words were analyzed by ALPNET and then the output was parsed to generate the word-root pairs. One list was extracted from a small corpus of Arabic text, called Zad. The Zad corpus is comprised of topics from a 14th century religious book called *Zad Al-Me'ad*. The list contained 9,606 words that ALPNET was able to analyze successfully. The original list was larger, but the words that ALPNET was unable to analyze were excluded. This list will be referred to as the ZAD list. The other list was extracted from the Linguistic Data Consortium (LDC) Arabic corpus containing AFP newswire stories (Graff & Walker, 2001). This list contained 562,684 words. Of these words, ALPNET was able to analyze 270,468 words successfully and failed to analyze 292,216 words. The subsets which ALPNET was able to analyze or failed to analyze will be referred to as the LDC-Pass and LDC-fail lists respectively. ALPNET failed to analyze words belonging to the following subsets:

- a. Named entities and Arabized words, which are words that are adopted from other languages: An example of these includes the words *krdtš* – كردتش (Karadish) and *dymwqrATyħ* – ديموقراطية (Democracy).
- b. Misspelled words
- c. Words with roots not in the root list: An example of that is the word *jwAĎ* – جواظ, (seldom used word meaning pompous).
- d. Words with templates not programmed into the finite state transducer. ALPNET uses a separate list of allowed templates for each root. These lists are not comprehensive. An example of that is the word *msylmħ* – مسيلمه (miniature of *mسلمħ* – مسلمة, a person who is safe or submitting).
- e. Words with complex prefixes or suffixes: An example of that is the word *bAlxrAfAt* – بالخرافات (with the superstitions). ALPNET was able to analyze *xrAfAt* – خرافات (superstitions) and *AlxrAfAt* – الخرافات (the superstitions).

It is noteworthy that whenever ALPNET successfully analyzed a word, the generated analyses were always correct.* In the cases when ALPNET provided more than one analysis for a word (thus more than one possible root), all combinations of the word and each of the possible roots were used for training. Although using all the combinations might have distorted the statistics, there was no automatic method for determining which word-root combinations to pick from the possible ones for a word. Also, using all the pairs had the effect of expanding the number of training examples on which the analyzer can be trained.

13.3.2 Training

Sebawai aligns characters in a word and the corresponding root, from the word-root pairs, using regular expressions to determine the prefix, suffix, and stem template. As in Table 13.6, given the pair (wktAbhm – وكتابههم, ktb – كتب), the training module would generate w – و as the prefix, hm – هم as the suffix, and CCAC as the stem template (C's represent the letters in the root). The module increases the observed number of occurrences of the prefix w – و, the suffix hm – هم, and the template “CCAC” by one. The module takes into account the cases where there are no prefixes or suffixes and denotes either of them with the symbol “#”.

Table 13.6. The decomposition of the word wktAbhm – وكتابههم with root ktb – كتب

Parts	Prefix	stem – CCAC – <i>فعال</i>			suffix
Root		k – ك	t – ت		b – ب
Word	w – و	k – ك	t – ت	A – ل	b – ب hm – هم

After that, the lists of prefixes, suffixes, and templates are read through to estimate probabilities by dividing the occurrence of each by the total number of word-root pairs. The probabilities being calculated are given for character strings S1 and S2 and template T as:

$$P(S1 \text{ begins a word}, S1 \text{ is a prefix}) = \frac{(\text{No. of words with prefix } S1)}{(\text{Total No. of training words})}$$

$$P(S2 \text{ begins a word}, S2 \text{ is a suffix}) = \frac{(\text{No. of words with suffix } S2)}{(\text{Total No. of training words})}$$

$$P(T \text{ is a template}) = \frac{(\text{No. of words with template } T)}{(\text{Total No. of training words})}$$

* A random set of a 100 words, for which ALPNET produced analyses, were manually examined to verify their correctness. ALPNET produced correct analyses for every word in the set.

Notice that Sebawai's stems are slightly different from standard stems, in that standard stems may have letters added in the beginning. Linguistics stems are often referred to in Arabic as $tf\text{c}y\text{A}t$ – تفعيلات or standard stem templates. For example, the template mCCwC has “m” placed before the root making “m” a part of the stem template. However, the training module of Sebawai has no prior knowledge of standard stem templates. Therefore, for the template “mCCwC,” the training module treated “m” as a part of the prefix list and the extracted template is “CCwC.”

13.3.3 Root Detection

Sebawai detects roots by reading in an Arabic word and attempts to generate feasible prefix-suffix-template combinations. The combinations are produced by generating all possible ways to break the word into three parts provided that the initial part is on the list of prefixes, the final part is in the list of suffixes, and the middle part is at least 2 letters long. The initial and final parts can also be null. Sebawai then attempts to fit the middle part into one of the templates. If the middle part fits into a template, the generated root is checked against a list of possible Arabic roots. The list of Arabic roots, which was extracted from a large Arabic dictionary, contained approximately 10,000 roots (Ibn Manzour, 2006). For example, the Arabic word AymAn – ايمان has the possible prefixes “#”, A – ا, and Ay – اي, and the possible suffixes “#”, n – ن, and An – ان. Table 13.7 shows the possible analyses of the word.

The stems that Sebawai deemed as not feasible were AymA – ايما and ym – يم. Although, AymA – ايما is not feasible, ym – يم is actually feasible (comes from the root ymm – يمم). This problem will be addressed in the following subsection. The possible roots are ordered according to the product of the probability that a prefix S1 would be observed, the probability that a suffix S2 would be observed, and the probability that a template T would be used as follows:

$$P(\text{root}) = P(S1 \text{ begins a word, } S1 \text{ is a prefix}) * P(S2 \text{ ends a word, } S2 \text{ is a suffix}) * P(T \text{ is a template})$$

Table 13.7. Possible analyses for the word AymAn – ايمان

Stem	Prefix	Template	Suffix	Root
AymAn – ايمان	#	CyCAC -- فيعال	#	Amn – أمن
ymAn – يمان	A – ا	CCAC – فعال	#	ymn – يمن
mAn – مان	Ay – اي	CCC – فعل	#	mAn – مان
Aym – أيم	#	CCC – فعل	An – ان	Aym – أيم
ymA – يما	A – ا	CCC – فعل	n – ن	ymA – يما

13.3.4 Extensions

- Handling Cases Where Sebawai Failed

As seen above, Sebawai deemed the stem $ym - يم$ not feasible, while in actuality the stem maps to the root $ymm - يمم$. The stem $ym - يم$ has only two letters because the third letter was dropped. The third letter is often dropped when it is identical to the second letter in the root. Sebawai initially failed to analyze all 2-letter stems. Two letter stems can be derived from roots with long vowels and where the second and third letters are the same.

The Arabic long vowels are $A - ا$, $y - ي$, and $w - و$. The weak letters are frequently substituted for each other in stem templates or dropped all together. For example, the word $qAl - قال$, has the root $qwl - قول$, or $qyl - قيل$, which would make the word mean ‘he said’ or ‘he napped’ respectively. Also, the stem $f - ف$ has the root $wfy - وفي$ where the letters $w - و$ and $y - ي$ are missing. To compensate for these problems, two letter stems were corrected by introducing new stems that are generated by doubling the last letter (to produce $ymm - يمم$ from $ym - يم$) and by adding weak letters before or after the stem. As for stems with a weak middle letter, new stems are introduced by substituting the middle letter with the other weak letters. For example, for $qAl - قال$, the system would introduce the stems $qwl - قول$ and $qyl - قيل$. This process over-generates potential roots. From the three potential roots $qAl - قال$, $qwl - قول$, and $qyl - قيل$, $qAl - قال$ is not a valid root and is thus removed (by comparing to the list of valid roots). To account for the changes, the following probabilities were calculated: (a) the probability that a weak letter would be transformed into another weak letter, (b) the probability that a two letter word would have a root with the second letter doubled (such as $ymm - يمم$), and (c) the probability that a two letter word was derived from a root by dropping an initial or trailing weak letter. The new probability of the root becomes:

$$P(\text{root}) = P(S1 \text{ begins a word, } S1 \text{ is a prefix}) * P(S2 \text{ ends a word, } S2 \text{ is a suffix}) * P(T \text{ is a template}) *$$

$$P(\text{letter substitution or letter addition})$$

- Simplifying Assumptions and Smoothing

The probabilities of stems, suffixes, and templates are assumed to be independent. The independence assumption is made to simplify the ranking, but is not necessarily a correct assumption because certain prefix-suffix combinations are not allowed. As for smoothing the prefix and suffix probabilities, Witten-Bell discounting was used (Jurafsky & Martin, 2000). The smoothing is necessary because many prefixes and suffixes were erroneously produced. This is a result of word-root pair letter alignment errors. Using this smoothing strategy, if a prefix or a suffix is observed only once, then it is removed from the respective list.

- Manual Processing

The list of templates was manually reviewed by an Arabic speaker (the first author) to insure the correctness of the templates. If a template was deemed not correct, it was removed from the list. Checking the list required approximately an hour.

- Particles

To account for particles, which are function words (equivalent to prepositions and pronouns in English) that are typically removed in retrieval applications, a list of Arabic particles was constructed with aid of *An-Nahw Ash-Shamil*, an Arabic grammar book (Abdul-Al-Aal, 1987). If the system matched a potential stem to one of the words on the particle list, the system indicated that the word is a particle. Note that particles are allowed to have suffixes and prefixes. A complete list of the particles is included in the distribution of *Sebawai* (Darwish, 2002).

- Letter Normalizations

The system employs a letter normalization strategy in order to account for spelling variations and to ease analysis. The first normalization deals with the letters ﻱ (*ya*) and ﻯ (*yalif maqSūra*). Both are normalized to $y - \text{ﻱ}$. The reason behind this normalization is that there is not a single convention for using $y - \text{ﻱ}$ or $\text{ﻱ} - \text{ﻯ}$ when either appears at the end of a word (Note that $Y - \text{ﻯ}$ only appears at the end of words). In the Othmani script of the Holy Qur'an for example, any $y - \text{ﻱ}$ is written as $\text{ﻱ} - \text{ﻯ}$ when it appears at the end of a word. The second normalization is that ﺀ (*hamza*), ﺀ (*A - alif*), ﺀ (*Ā - alif mamduuda*), ﺀ (*Â - alif with hamza on top*), ﺀ (*â - hamza on w*), ﺀ (*Ă - alif with hamza on the bottom*), and ﺀ (*ÿ - hamza on ya*) are normalized to $A - \text{ﺀ}$ (*A - alif*). The reason for this normalization is that all forms of hamza are represented in dictionaries as one in root form, and people often misspell different forms of *alif*.

Lastly, words are stripped of all diacritics.

13.3.5 Evaluation

To evaluate *Sebawai*, it was compared to ALPNET. The two analyzers were compared to test *Sebawai*'s highest ranked analysis and the coverage of *Sebawai* and ALPNET. The experiments performed involved training *Sebawai* using the LDC-pass word-root list and testing on the ZAD list. Of the 9,606 words in the ZAD list, *Sebawai* analyzed 9,497 words and failed on 112. For the roots generated by *Sebawai*, the top generated root was automatically compared to the roots generated by ALPNET. If the root is on the list, it is considered correct. Using this method, 8,206 roots were considered correct. However, this automatic evaluation had two flaws:

1. The number of Arabic roots in ALPNET's inventory is only 4,500 roots while the number of roots used by *Sebawai* is more than 10,000. This could lead to false negatives in the evaluation.

- ALPNET often under-analyzes. For example the word *fy* – في could be the particle *fy* – في or could be a stem with the root *fyy* – فيي. ALPNET only generates the particle *fy* – في. This also could lead to false negatives.

Therefore manual examination of rejected analyses was necessary. However, due to the large number of rejected analyses, 100 rejected analyses from the automatic evaluation were randomly selected for examination to estimate the shortfall of the automatic evaluation. Of the 100 rejected roots, 46 were correct and 54 were incorrect. Figure 13.2 and Table 13.8 present a summary of the results.

For the LDC-Fail list, Sebawai analyzed 128,169 (43.9%) words out of 292,216 words. To verify the correctness of Sebawai’s analyses, 100 analyses were taken at random from the list for manual examination. Of the 100 analyses, 47 were actually analyzed correctly. Extrapolating from the results of the manual examination, Sebawai successfully analyzed approximately 21% of the words in the LDC-Fail list. Table 13.9 presents a summary of the results.

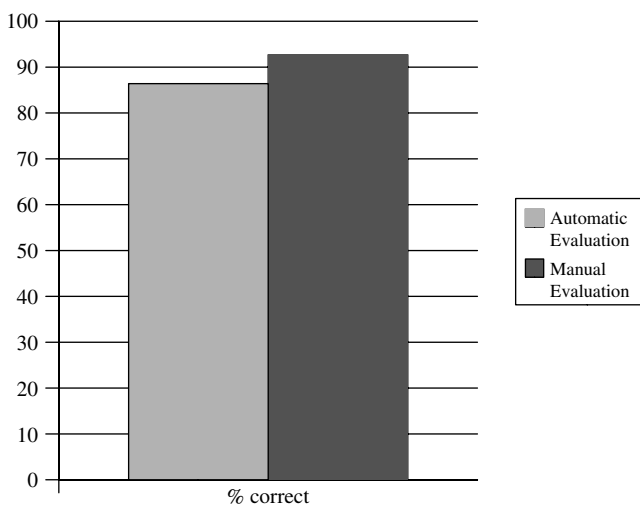


Fig. 13.2. A summary of the results for the correctness of Sebawai’s analysis

Table 13.8. A summary of the results for the correctness of Sebawai’s analysis

No. of Words	No. of Failures	No. Correct – Automatic Evaluation	No. Correct – Manual Evaluation
9, 606	112 (1.2%)	8,206 (86.4)	8,800 (92.7%)

Table 13.9. Summary of the results for the LDC-Fail list

No. of Words	No. of Analyzed Words	Estimate of Correctly Analyzed
292216	128,169 (43.9%)	58,000 (21%)

The evaluation clearly shows the effectiveness of Sebawai in achieving its two primary goals. Namely, Sebawai provides ranking to the possible analysis and expands the coverage beyond ALPNET. For words that ALPNET was able to analyze, Sebawai ranked the correct analysis first for nearly 93% of the words. Further, the analyzer was able to correctly analyze 21% of the words that ALPNET was unable to analyze. Also, due to the fact that prefixes, suffixes, and templates were derived automatically, Sebawai was developed very rapidly.

13.3.6 Shortcomings of Sebawai

Since analysis is restricted to a choice from a fixed set of roots, Sebawai does not stem Arabized words and named entities. For example, the English word Britain is transliterated as bryTAnyA – بريطانيا. From bryTAnyA – بريطانيا, some of the words that can be generated are bryTAny – بريطاني (British), AlbryTAny – البريطاني (the British), and AlbryTAnyyn – البريطانيين (Englishmen). Sebawai is unable to analyze 1-letter Arabic words, which have 3-letter roots. For example, the word q – ق means “protect (in the form of command).” Since they are very rare, they may not appear in the training set. Sebawai is unable to analyze individual Arabic words that constitute complete sentences. For example, the word AnlzmkmwhA – أنلزمكموها means “will we forcefully bind you to it?” These also are rare and may not appear in a training set.

13.4 Light Stemming

To build the light stemmer, Al-Stem, the lists of prefixes and suffixes generated in the process of training Sebawai and their corresponding probabilities were examined. If a prefix or a suffix had a probability of being an affix above 0.5, it was considered a candidate for building Al-Stem. The list of prefix and suffix candidates was manually examined in consultation with Leah Larkey from the University of Massachusetts at Amherst and some of the affixes were removed based on intuition and knowledge of Arabic.

The final lists of prefixes and suffixes are as follows:

- Prefixes: wAl – ولا, fAl – فلا, bAl – بلا, bt – بت, yt – يت, lt – لت, mt – مت, wt – وت, st – ست, nt – نت, bm – بم, lm – لم, wm – وم, km – كم, fm – فم, Al – ال, ll – لل, wy – وي, ly – لي, sy – سي, fy – في, wA – وا, fA – فا, lA – لا, and bA – با.

- Suffixes: At – ات, wA – وا, wn – ون, wh – وه, An – ان, ty – تي, th – ته, tm – تم, km – كم, hm – هم, hn – هن, hA – ها, yh – يه, tk – تك, nA – نا, yn – ين, yh – يه, h – ه, y – ي, A – ا.

Since the result of light stemming may or not be a real stem, no effort was made to evaluate the correctness of the stemming.

13.5 Evaluating Sebawai and AI-Stem in IR

IR experiments were done on the LDC LDC2001T55 collection, which was used in the Text REtrieval Conference (TREC) 2002 cross-language track. For brevity, the collection is referred to as the TREC collection. The collection contains 383,872 articles from the Agence France Press (AFP) Arabic newswire. Fifty topics were developed cooperatively by the LDC and the National Institute of Standards and Technology (NIST), and relevance judgments were developed at the LDC by manually judging a pool of documents obtained from combining the top 100 documents from all the runs submitted by the participating teams to TREC's cross-language track in 2002. The number of known relevant documents ranges from 10 to 523, with an average of 118 relevant documents per topic (Oard & Gey, 2002). This is presently the best available large Arabic information retrieval test collection. The TREC topic descriptions include a title field that briefly names the topic, a description field that usually consists of a single sentence description, and a narrative field that is intended to contain any information that would be needed by a human judge to accurately assess the relevance of a document (Gey & Oard, 2001). Two types of queries were formed from the TREC topics:

- a. The title and description fields (td). This is intended to model the sort of statement that a searcher might initially make when asking an intermediary such as a librarian for help.
- b. The title field only (t). The title field in recent TREC collections is typically designed as a model for Web queries, which typically contain only 2 or 3 words.

Experiments were performed for each query length with the following index terms:

- w: words.
- ls: lightly stemmed words, obtained using AI-Stem.
- Two ways of obtaining stems:
 - s: top stem found by the Sebawai morphological analyzer (Darwish, 2002).
 - s-ma: top ranking stem found by Sebawai, produced also by ALPNET (Beesley, 1996; Beesley et al., 1989), if ALPNET produced an analysis; otherwise the top stem found by Sebawai. Recall that for words that it can analyze, ALPNET produces an unranked set of analyses, but it fails to produce an analysis more often than Sebawai.
- Two ways of obtaining roots:
 - r: top root found by the Sebawai morphological analyzer (Darwish, 2002).
 - r-ma: top ranking root found by Sebawai, produced also by ALPNET, if ALPNET produced an analysis; otherwise the top root found by Sebawai.

For the experiments, a vector space model information retrieval system called PSE that uses the Okapi BM-25 term weights was used (Robertson & Sparck Jones, 1997). To observe the effect of alternate indexing terms mean uninterpolated average precision was used as the measure of retrieval effectiveness. To determine if the difference between results was statistically significant, a Wilcoxon signed-rank test, which is a nonparametric significance test for correlated samples, was used with p values less than or equal to 0.05 to claim significance.

Figure 13.3 summarizes the results of these runs, and Tables 13.10 and 13.11 present the statistical significance test results for the t and td queries respectively. These results indicate that light stems did not only statistically significantly yield better results than words, but light stems yielded better results than stems and roots. Also, filtering Sebawai's candidates using ALPNET hurt retrieval considerably. This could be due to ALPNET frequently producing analyses that did not include Sebawai's top ranked analysis and thus randomizing the choice of analysis to the detriment of retrieval effectiveness. Recall that ALPNET produces analysis in random order. As indicated earlier, some early work with small test collections (Al-Kharashi & Evens, 1994; Hmeidi et al., 1997) suggested that roots were a better choice than stems, but the experiments presented here found just the opposite. One possible explanation for this is that earlier test collections contained at most a few hundred documents, and scaling up the size of the collection by several orders of magnitude might reward the choice of less ambiguous terms. An alternative explanation is that Sebawai's morphological analysis might not be sufficiently accurate.

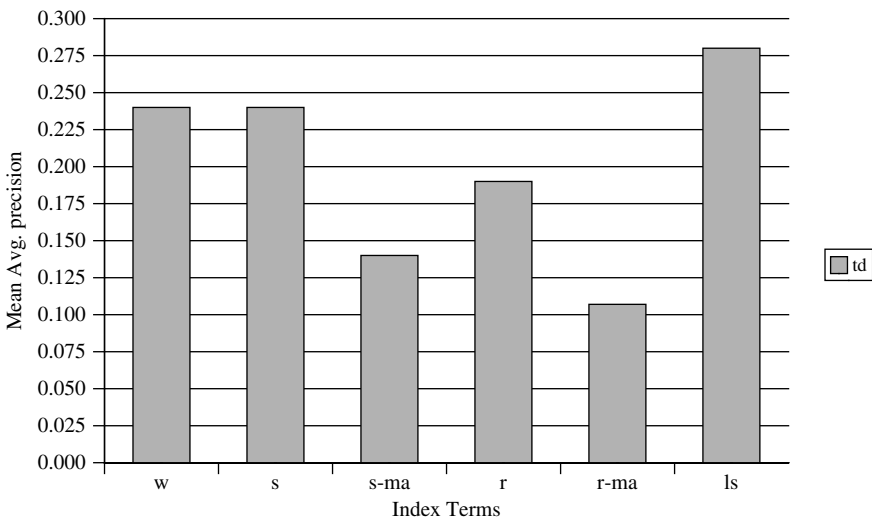


Fig. 13.3. Comparing index terms on the TREC collection for title only (t) and title+description queries

Table 13.10. Comparing index terms using title only queries on the TREC collection using the p values of the Wilcoxon signed-ranked test – the cell is shaded if difference is statistically significant

<i>w</i>	<i>S</i>	<i>r</i>	<i>ls</i>	
	0.24	0.39	0	<i>w</i>
		0.01	0	<i>s</i>
			0	<i>r</i>
				<i>ls</i>

Table 13.11. Comparing index terms using title+description queries on the TREC collection using the p values of the Wilcoxon signed-ranked test – the cell is shaded if difference is statistically significant

<i>w</i>	<i>s</i>	<i>r</i>	<i>ls</i>	
	0.92	0.38	0	<i>w</i>
		0.02	0	<i>s</i>
			0	<i>r</i>
				<i>ls</i>

13.6 Conclusion

This chapter presented an adaptation of existing Arabic morphological analysis techniques to make them suitable for the requirements of IR applications by leveraging corpus statistics. As a result of the adaptation, Sebawai, a new freely distributable Arabic morphological analyzer, and Al-Stem, an Arabic light stemmer were constructed. Al-Stem was used extensively by many research groups for comparative Arabic IR evaluations (Chen & Gey, 2002; Darwish & Oard 2002a; Fraser et al., 2002; Larkey et al., 2002; Oard & Gey, 2002).

The morphological analysis techniques described here could benefit from better word models that incorporate statistics on legal prefix-suffix combination (currently Sebawai assumes independence between prefixes and suffixes) and sentence level models that incorporate context to improve ranking. All these techniques can potentially be used in developing morphological analyzers for other morphologically similar languages such as Hebrew.

References

- Abdul-Al-Aal, A. (1987). *An-Nahw Ashamil*. Cairo, Egypt: Maktabat Annahda Al-Masriya.
- Abu-Salem, H., Al-Omari, M. & Evens, M. (1999). Stemming Methodologies Over Individual Query Words for Arabic Information Retrieval. *Journal of the American Society for Information Science and Technology*, 50(6), 524–529.
- Ahmed, M. (2000). *A Large-Scale Computational Processor of Arabic Morphology and Applications*. Faculty of Engineering, Cairo University, Cairo, Egypt.
- Aljlal, M., Beitzel, S., Jensen, E., Chowdhury, A., Holmes, D., Lee, M., Grossman, D. & Frieder, O (2001). IIT at TREC-10. In *Proceedings of the Tenth Text REtrieval Conference* (pp.265–274), Gaithersburg, MD. <http://trec.nist.gov/pubs/trec10/papers/IIT-TREC10.pdf>
- Al-Kharashi, I. & Evens, M. (1994). Comparing Words, Stems, and Roots as Index Terms in an Arabic Information Retrieval System. *Journal of the American Society for Information Science and Technology*, 45(8), 548–560.
- Antworth, E. (1990). PC-KIMMO: a two-level processor for morphological analysis. In *Occasional Publications in Academic Computing*. Dallas, TX: Summer Institute of Linguistics.
- Beesley, K. (1996). Arabic Finite-State Morphological Analysis and Generation. In *Proceedings of the International Conference on Computational Linguistics (COLING-96, vol. 1, pp. 89–94)*.
- Beesley, K., Buckwalter, T. & Newton, S. (1989). Two-Level Finite-State Analysis of Arabic Morphology. In *Proceedings of the Seminar on Bilingual Computing in Arabic and English*, Cambridge, England.
- Chen, A. & Gey, F. (2001). Translation Term Weighting and Combining Translation Resources in Cross-Language Retrieval. In *Proceedings of the Tenth Text REtrieval Conference* (pp. 529–533), Gaithersburg, MD. http://trec.nist.gov/pubs/trec10/papers/berkeley_trec10.pdf
- Chen, A. & Gey, F. (2002). Building an Arabic Stemmer for Information Retrieval. In *Proceedings of the Eleventh Text REtrieval Conference*, Gaithersburg, MD. <http://trec.nist.gov/pubs/trec11/papers/ucalberkeley.chen.pdf>
- Darwish, K. (2002). Building a Shallow Arabic Morphological Analyzer in One Day. In *Proceedings of the ACL Workshop on Computational Approaches to Semitic Languages* (pp. 1–8). Philadelphia, PA.
- Darwish, K. & Oard, D. (2002a). CLIR Experiments at Maryland for TREC 2002: Evidence Combination for Arabic-English Retrieval. In *Proceedings of the Eleventh Text REtrieval Conference*, Gaithersburg, MD. <http://trec.nist.gov/pubs/trec11/papers/umd.darwish.pdf>
- Darwish, K. & Oard, D. (2002b). Term Selection for Searching Printed Arabic. In *Proceedings of the Special Interest Group on Information Retrieval Conference (SIGIR, pp. 261–268)*, Tampere, Finland.
- Fraser, A., Xu, J. & Weischedel, R. (2002). TREC 2002 Cross-lingual Retrieval at BBN. In *Proceedings of the Eleventh Text REtrieval Conference*, Gaithersburg, MD. <http://trec.nist.gov/pubs/trec11/papers/bbn.xu.cross.pdf>
- Gey, F. & Oard, D. (2001). The TREC-2001 Cross-Language Information Retrieval Track: Searching Arabic Using English, French or Arabic Queries. In *Proceedings of the Tenth Text REtrieval Conference*, Gaithersburg, MD. <http://trec.nist.gov/pubs/trec10/papers/clirtrack.pdf>
- Goldsmith, J. (2000). *Unsupervised Learning of the Morphology of a Natural Language*. Retrieved from <http://humanities.uchicago.edu/faculty/goldsmith/>

- Graff, D. & Walker, K. (2001). *Arabic Newswire Part 1*. Linguistic Data Consortium, Philadelphia. LDC catalog number LDC2001T55 and ISBN 1-58563-190-6.
- Hmeidi, I., Kanaan, G. & Evens, M. (1997). Design and Implementation of Automatic Indexing for Information Retrieval with Arabic Documents. *Journal of the American Society for Information Science and Technology*, 48(10), 867–881.
- Ibn Manzour (2006). *Lisan Al-Arab*. Retrieved from <http://www.muhammadith.org/>
- Jurafsky, D. & Martin, J. (2000). *Speech and Language Processing*. Saddle River, NJ: Prentice Hall.
- Kiraz, G. (1998). Arabic Computational Morphology in the West. In *Proceedings of The 6th International Conference and Exhibition on Multi-lingual Computing*, Cambridge.
- Koskenniemi, K. (1983). *Two Level Morphology: A General Computational Model for Word-form Recognition and Production*. Department of General Linguistics, University of Helsinki.
- Larkey, L., Allen, J., Connell, M. E., Bolivar, A. & Wade, C. (2002). UMass at TREC-2002: Cross Language and Novelty Tracks. In *Proceedings of the Eleventh Text REtrieval Conference*, Gaithersburg, MD. <http://trec.nist.gov/pubs/trec11/papers/umass.wade.pdf>
- Larkey, L., Ballesteros, L. & Connell, M. (2002). Improving Stemming for Arabic Information Retrieval: Light Stemming and Co-occurrence Analysis. In *Proceedings of the Special Interest Group on Information Retrieval (SIGIR)*, pp. 275–282), Tampere, Finland.
- Mayfield, J., McNamee, P., Costello, C., Piatko, C. & Banerjee, A. (2001). JHU/APL at TREC 2001: Experiments in Filtering and in Arabic, Video, and Web Retrieval. In *Proceedings of the Tenth Text REtrieval Conference*, Gaithersburg, MD. <http://trec.nist.gov/pubs/trec10/papers/jhuapl01.pdf>
- Oard, D. & Gey, F. (2002). *The TREC 2002 Arabic/English CLIR Track*. In *Proceedings of the Eleventh Text REtrieval Conference*, Gaithersburg, MD. <http://trec.nist.gov/pubs/trec11/papers/OVERVIEW.gey.ps.gz>
- Robertson, S. & Sparck Jones, K. (1997). *Simple Proven Approaches to Text Retrieval*. Cambridge University Computer Laboratory.
- Xu, J., Fraser, A. & Weischedel, R. (2001). Cross-Lingual Retrieval at BBN. In *Proceedings of the Tenth Text REtrieval Conference* (pp. 68–75), Gaithersburg, MD. <http://trec.nist.gov/pubs/trec10/papers/BBNTREC2001.pdf>