12

Light Stemming for Arabic Information Retrieval

Leah S. Larkey[1], Lisa Ballesteros[2] and Margaret E. Connell[3]

[1] *Chiliad Publishing, 44 Belchertown Rd, Amherst, MA 01002*
LarkeyLeah@gmail.com

[2] *Computer Science Dept., Mt. Holyoke College, South Hadley, MA 01075*
lballest@mtholyoke.edu

[3] *Univ. of Massachusetts, Dept. of Computer Science, Amherst, MA 01003*
connell@cs.umass.edu

Abstract: Computational Morphology is an urgent problem for Arabic Natural Language Processing, because Arabic is a highly inflected language. We have found, however, that a full solution to this problem is not required for effective information retrieval. Light stemming allows remarkably good information retrieval without providing correct morphological analyses. We developed several light stemmers for Arabic, and assessed their effectiveness for information retrieval using standard TREC data. We have also compared light stemming with several stemmers based on morphological analysis. The light stemmer, light10, outperformed the other approaches. It has been included in the Lemur toolkit, and is becoming widely used Arabic information retrieval

12.1 Introduction

The central problem of Information Retrieval (IR) is to find documents that satisfy a user's information need, usually expressed in the form of a query. This active research area has seen great progress in recent decades, which everyone has experienced in searching the internet. Initially, most IR research was carried out in English and fueled by the annual Text Retrieval Conferences (TREC) sponsored by NIST (the National Institute of Standards and Technology). NIST has accumulated large amounts of standard data (text collections, queries, and relevance judgments) so that IR researchers can compare their techniques on common data sets. More recently, IR research involving other languages has flourished. TREC now includes multilingual data and in recent years, other organizations sponsor similar annual evaluations for European languages (CLEF) and Asian languages (NTCIR) (Chinese, Japanese, and Korean). Arabic began to be included in the TREC cross-lingual track in 2001, and in the TDT (topic detection and tracking) evaluations in 2001 [47]. The availability of standard Arabic data sets from the NIST and the Linguistic Data Consortium (LDC) has in turn spurred a huge

A. Soudi, A. van den Bosch and G. Neumann (eds.), Arabic Computational Morphology, 221–243.

acceleration in progress in information retrieval and other natural language processing applications involving Arabic.

Any discussion of multilingual retrieval requires a distinction between monolingual retrieval in multiple languages, and cross-lingual or cross-language retrieval. In monolingual retrieval, queries are issued in the same language as the documents in the collection being searched. In cross-lingual retrieval, queries are issued in a different language than the documents in the collection. A central problem in both monolingual and cross-lingual streams of IR research is the *vocabulary mismatch* problem. The same information need can be expressed using different terminology (the disease *bilharzia* is also called *schistosomiasis*), or a key term can have different spellings (e.g. *theater* versus *theatre*), or may be inflected differently in the query than in the relevant documents. We discuss in Section 12.1.1 the kinds of variability that lead to a particularly acute vocabulary mismatch problem in Arabic information retrieval. Morphological variation in IR has generally been handled by stemming, an unsophisticated but effective approach to morphology which we discuss in Section 12.1.2.

12.1.1 Arabic Morphology and Orthography

The morphological and orthographic complexity of Arabic (see Chapter 3 of this volume) makes it particularly difficult to develop natural language processing applications for Arabic information retrieval.

Distributional analyses of Arabic newspaper text show empirically that there is more lexical variability in Arabic than in the European languages for which most IR and NLP work has been performed. Arabic text has more words occurring only once and more distinct words than English text samples of comparable size.[1] The token to type ratio (mean number of occurrences over all distinct words in the sample) is smaller for Arabic texts than for comparably sized English texts [28].

For information retrieval, this abundance of forms, lexical variability, and orthographic variability, all result in a greater likelihood of mismatch between the form of a word in a query and the forms found in documents relevant to the query. In cross-language retrieval there is an additional serious mismatch problem between query terms and the forms found in the bilingual dictionaries that are used in cross-language retrieval.

To deal with this variability, most morphological analyzers attempt to insert missing short vowels and other diacritics. However for information retrieval, the opposite approach is more typical. Text is normalized by removing many diacritics and short vowels.

[1] We use the term *word* in simple sense of text segmented at white space or punctuation, without any morphological analysis.

12.1.2 Stemming in Information Retrieval

Stemming is another one of many tools besides normalization that is used in information retrieval to combat this vocabulary mismatch problem. Stemmers equate or *conflate* certain variant forms of the same word like (*paper, papers*) and (*fold, folds, folded, folding…*). In this work, we use the term *stemming* to refer to any process which conflates related forms or groups forms into equivalence classes, including but not restricted to suffix stripping. In this section we review general approaches to stemming over many languages. We focus on Arabic in the next section. Most approaches fall into two classes: affix removal and statistical stemming.

12.1.2.1 Affix Removal

In English and many other western European languages, stemming is primarily a process of suffix removal [41][50]. Such stemmers do not conflate irregular forms such as (*goose, geese*) and (*swim, swam*). These stemmers are generally tailored for each specific language. Their design requires some linguistic expertise in the language and an understanding of the needs of information retrieval. Stemmers have been developed for a wide range of languages including Malay [54], Latin [29], Indonesian [8], Swedish [12], Dutch [35], German [44], French [45], Slovene [49], and Turkish [21]. The effectiveness of stemming across languages is varied and influenced by many factors. A reasonable summary is that stemming doesn't hurt retrieval; it either makes little difference or it improves effectiveness by a small amount [31]. Stemming is considered to aid recall more than precision [35]. That is, stemming allows a search engine to find more relevant documents, but may not improve its ability to rank the best documents at the top of the list. Stemming appears to have a larger positive effect when queries and/or documents are short [36], and when the language is highly inflected [48][49], suggesting that stemming should improve Arabic information retrieval.

12.1.2.2 Statistical Techniques

Statistical methods can provide a more language-independent approach to conflation. Related words can be grouped based on various string-similarity measures. Such approaches often involve n-grams. Equivalence classes can be formed from words that share word-initial letter n-grams or a threshold proportion of n-grams throughout the word, or by refining these classes with clustering techniques. This kind of statistical stemming has been shown to be effective for many languages, including English, Turkish, and Malay [21][23][24][46].

Statistical techniques have widely been applied to automatic morphological analysis in computational linguistics [9][17][22][26][27][30][32][34]. For example, Goldsmith finds the best set of frequently occurring stems and suffixes using an information theoretic measure [26]. Oard et al. consider the most frequently occurring word-final n-grams (1, 2, 3, and 4-grams) to be suffixes [46].

Stem classes can also be built or refined using co-occurrence analysis, which Xu and Croft proposed as a promising language-independent approach to stemming [55]. Stemmers make two kinds of errors. Weak stemmers fail to conflate related forms that should be grouped together. Strong stemmers tend to form larger stem classes in which unrelated forms are erroneously conflated. Most stemmers fall between these two extremes and make both kinds of errors. Xu and Croft employ a corpus analysis approach which is particularly suited to splitting up stem classes created by strong stemmers. The stem-classes are reclustered based on a co-occurrence measure, which is language independent in that it can be applied to any set of stem classes. Xu and Croft applied their technique to effectively stem English and Spanish and obtained two important results. First, one can refine an already good stemmer by co-occurrence analysis and improve retrieval effectiveness. Second, one can start with a strong crude stemmer like an n-gram stemmer and use co-occurrence analysis to yield stem classes that work as well as a sophisticated stemmer. They demonstrated an improvement in retrieval effectiveness for English and Spanish after clustering conventional and n-gram based stem classes.

12.1.3 Stemming and Morphological Analysis in Arabic for Information Retrieval

The factors described in Section 12.1.1 make Arabic very difficult to stem. The issue of whether roots or stems are the desired level of analysis for IR has been one complication that has given rise to additional approaches to stemming for Arabic besides affix removal and the statistical stemming approaches described above. Other approaches include manual dictionary construction, morphological analysis, and new statistical methods involving parallel corpora.

12.1.3.1 Manual Construction of Dictionaries

Early work on Arabic stemming used manually constructed dictionaries. Al-Kharashi and Evens worked with small text collections, for which they manually built dictionaries of roots and stems for each word to be indexed [4]. This approach is obviously impractical for realistic sized corpora.

12.1.3.2 Affix Removal

The affix removal approach is generally called *light stemming* when applied to Arabic, referring to a process of stripping off a small set of prefixes and/or suffixes, without trying to deal with infixes, or recognize patterns and find roots. Light stemming was used for Arabic by some authors without details in work prior to ours [3][18]. No explicit lists of strippable prefixes and/or suffixes or algorithms had been published at the time we did this research. Our light stemmer, *light10*, strips off initial و (*w*=and), some prepositions, definite articles (ال، وال، بال، كال، فال، لل) and suffixes (ها، ان، ات، ون، ين، يه، ية، ه، ة، ي). (More detail can be found in Section 12.2.2.) It was designed to strip off strings that were frequently found as prefixes or suffixes, but infrequently found at the beginning or ending of stems. It was not intended to be exhaustive. Darwish introduced the *Al-Stem* light stemmer at TREC 2002 [16], and demonstrated that it was less effective than light10. Chen and Gey [13] introduced a light stemmer similar to light10, but that removed more prefixes and suffixes. It was shown to be more effective than Al-Stem, but was not directly compared to light10.

Although light stemming can correctly conflate many variants of words into large stem classes, it can fail to conflate other forms that should go together. For example, broken (irregular) plurals for nouns and adjectives do not get conflated with their singular forms, and past tense verbs do not get conflated with their present tense forms, because they retain some affixes and internal differences. In spite of the simplicity and shortcomings of light stemming, more sophisticated approaches have not proven to be more effective for information retrieval.

12.1.3.3 Statistical Stemming

Although n-gram systems have been used for many different languages, one would not expect them to perform well on infixing languages like Arabic. However, Mayfield et al. have developed a system that combines word-based and 6-gram based retrieval, which performs remarkably well for many languages [43] including Arabic [42].

De Roeck and Al-Fares [18] used clustering on Arabic words to find classes sharing the same root. Their clustering was based on morphological similarity, using a string similarity metric tailored to Arabic morphology, which was applied after removing “a small number of obvious affixes.” They evaluated the technique by comparing the derived clusters to “correct” classes. They did not assess the performance in an information retrieval context.

We applied Xu and Croft’s co-ocurrence method to Arabic [55]. We assumed that initial n-gram based stem classes were probably not the right starting point for languages like Arabic. However, co-occurrence or other clustering techniques can be applied to Arabic without using n-grams. Instead, we formed classes of words that mapped onto the same string if vowels were removed, and used co-occurrence measures to split these classes further. The co-occurrence method did not work as well for Arabic as it did for English and Spanish. It did not produce a stemmer that worked as well as light10. We found that while one could improve a mediocre stemmer with this technique, its effectiveness was still far from the level attained by a high-quality stemmer like light10. And the light10 stemmer could not be further improved by co-occurrence analyses. Perhaps this is because of the big stemming effect in Arabic compared to English or Spanish.

A promising new class of statistical stemmers makes use of parallel corpora. Chen and Gey [13] used a parallel English-Arabic corpus and an English stemmer to cluster Arabic words into stem classes based on their mappings to English stem classes. Rogati, McCarley, and Yang [51] use a statistical machine translation approach that learns to split words into prefix, stem, and suffix by training on a small hand annotated training set and using a parallel corpus These approaches work well considering how automated they are, but they are not as effective in an IR evaluation as a good light stemmer.

12.1.3.4 Morphological Analysis

It is often assumed that stemming is just a quick and dirty way to approximate morphological analysis, and that the best way to stem would be to perform a correct morphological analysis and then use some valid morphological unit for indexing and retrieval. For Arabic, this unit has often been thought to be the root. Several morphological analyzers have been developed for Arabic [2][6][7][15] [33][19] but few have received a standard IR evaluation Most such morphological analyzers find the root, or any number of possible roots for each word. A morphological analyzer called Sebawai, developed by Kareem Darwish [14] was used by some of the TREC participants in 2001 [15][33], but it was not directly compared with light stemming.

In our earlier research we evaluated a simple morphological analyzer from Khoja and Garside [33], which first peels away layers of prefixes and suffixes, then checks a list of patterns and roots to determine whether the remainder could be a known root with a known pattern applied. If so, it returns the root. Otherwise, it returns the original word, unmodified. This system also removes terms that are found on a list of 168 Arabic stop words. It was almost as effective as light stemming, but tended to fail on foreign words, which it left unchanged rather than removing definite articles and obvious affixes. Taghva, Elkhoury, and Coombs [53] have developed a system that finds Arabic roots somewhat like Khoja's approach, but without using a root dictionary or lexicon, and which performs as well as a light stemmer.

Tim Buckwalter's morphological analyzer [10] is different from the others in that it returns stems rather than roots. It is based on a set of lexicons of Arabic stems, prefixes, and suffixes, with truth tables indicating their legal combinations. The BBN group used this table-based stemmer in TREC-2001 [56], but did not compare it with light stemming. The Buckwalter stemmer is now available from LDC [39], and is evaluated as a stemmer in the present study. Finally, Diab, Hacioglu, and Jurafsky [20] developed a set of tools for Arabic morphological analyses which learn tokenization, lemmatization, part-of-speech assignment, and phrase chunking, automatically using SVM (support vector machines), a machine learning categorization tool. Their tools are trained on a sample of the Arabic Tree Bank [40], which is a portion of the AFP database which has been analyzed by the Buckwalter morphological analyzer, and hand-corrected. They claim above 99% accuracy on tokenization, and 95.49% accuracy on POS tagging. We derive some stemmers from these tools, as part of the present study.

Early published comparisons of stems versus roots for information retrieval have claimed that roots are superior to stems, based on small, nonstandard test sets [1][4]. Recent work at TREC has found no consistent differences between roots and stems [15]. We found a small increase in effectiveness when we combined roots and stems [38]. However, we feel that roots versus stems is not the most interesting question to investigate. As this book makes clear, morphological analysis of Arabic is now an active research area, and many systems are being developed to return more complete analyses of Arabic words. A more interesting question is how to use morphological analysis to aid information retrieval, and in particular, to aid stemming. The new work in this chapter attempts to use morphological analysis to get to something better than a root for indexing.

The present study expands upon work we published at SIGIR in 2002 [37]. At that time, we developed several light stemmers, compared their effectiveness on an IR task with each other and with that of a morphological analyzer available at that time. We also experimented with a co-occurrence approach to improving stemming. The present research expands that study in several ways. First, the light stemmer we eventually settled upon (and used in TREC 2002) was slightly different from the best one reported at SIGIR. We compare it with the other stemmers here. Second, we use 75 queries from TREC 2001 and TREC 2002 to evaluate stemmers here, providing results that are more reliable than the 25 queries from TREC 2001 used in the previous study. Third, we evaluate stemming approaches based on two morphological analyzers that were not available when the earlier study was carried out.

12.2 Review of 2002 Stemming Experiments

In this section we review the light stemming experiments from the SIGIR article, but include in addition the modified stemmer, light10. This set of experiments was carried out using the TREC 2001 corpus and queries.

12.2.1 Experimental Method

The TREC-2001 Arabic corpus, also called the AFP_ARB corpus, consists of 383,872 newspaper articles in Arabic from *Agence France Presse*. This fills up almost a gigabyte in UTF-8 encoding as distributed by the Linguistic Data Consortium. There were 25 topics with relevance judgments, available in Arabic, French, and English, with *Title*, *Description*, and *Narrative* fields [25]. We used the Arabic titles and descriptions as queries in monolingual experiments, and the English titles and descriptions in cross-language experiments.

Corpus and queries were converted to CP1256 encoding and indexed using an in-house version of the INQUERY retrieval engine [11]. Arabic strings were treated as a simple string of bytes, regardless of how they would be rendered on the screen. Text was broken up into words at any white space or punctuation characters, including Arabic punctuation. Stop words were removed, using a stop word list from Khoja [33]. Words of one-byte length (in CP1256 encoding) were not indexed. The experiments reported here used INQUERY for retrieval.

Except for the *raw* condition, in which no normalization or stemming was used, the corpus and queries were normalized according to the following steps:

- Remove punctuation
- Remove diacritics (primarily weak vowels). Some entries contained weak vowels, in particular, the dictionaries used in cross-language experiments. Removal made everything consistent.
- Remove non letters
- Replace آ , إ , and أwith ا
- Replace final ى with ي
- Replace final ة with ه

For the normalized conditions and the stemming conditions, we normalized and stemmed all tokens before indexing the corpus, and normalized and stemmed the queries with the same stemmer for retrieval. Arabic queries were expanded using the technique of local context analysis, adding 50 terms from the top-10 documents, as described in detail in [38]. Expansion was performed in order to show the ultimate level of performance attainable using the stemmers in the context of our whole system.

12.2.2 Light Stemmers

Our guiding principle in designing the light stemmers was heuristic. A light stemmer is not dictionary driven, so it cannot apply a criterion that an affix can be removed only if what remains is an existing Arabic word. In fact, we suspect that part of the success of such stemmers is that they can blindly work on words even if they are not found in a word list. The attempt was to try to remove strings that would be found reliably as affixes far more often than they would be found as the beginning or end of an Arabic stem without affixes. We also benefited from discussions with some colleagues at TREC-2001, particularly M. Aljlayl. We tried several versions of light stemming, all of which followed the same steps:

1. Remove و ("and") for light2, light3, and light8, and light10 if the remainder of the word is 3 or more characters long. Although it is important to remove و, it is also problematic, because many common Arabic words begin with this character, hence the stricter length criterion here than for the definite articles.
2. Remove any of the definite articles if this leaves 2 or more characters.
3. Go through the list of suffixes once in the (right to left) order indicated in Table 12.1, removing any that are found at the end of the word, if this leaves 2 or more characters.

Table 12.1. Strings removed by light stemming

	Remove prefixes	Remove Suffixes
Light1	ال، وال، بال، كال، فال	none
Light2	ال، وال، بال، كال، فال، و	none
Light3	“	ه، ة
Light8	“	ها، ان، ات، ون، ين، يه، ية، ه، ة، ي
Light10	ال، وال، بال، كال، فال، لل، و	“

The strings to be removed are listed in Table 12.1. The “prefixes” are actually prepositions, definite articles and a conjunction. The light stemmers do not remove any strings that would be considered Arabic prefixes.

12.2.3 Results of Monolingual Stemmer Comparisons

Figure 12.1 shows precision at 11 recall points for the primary stemmers tested. *Raw* means no normalization or stemming. *Norm* refers to normalization with no stemming. *Light1, Light2, Light3, Light8, and Light10* refer to the light stemmers described above.

Table 12.2 shows uninterpolated average precision for the basic stemmers. For raw, normalized, and light stemming conditions performance is better with each successive increment in degree of stemming. Each of these increments is statistically significant except light10 versus light8.[2] As these results indicate, light stemming is remarkably effective. Light10 has become widely used, and has been included in the Lemur toolkit, a set of software tools for research in language modeling and information retrieval [5].

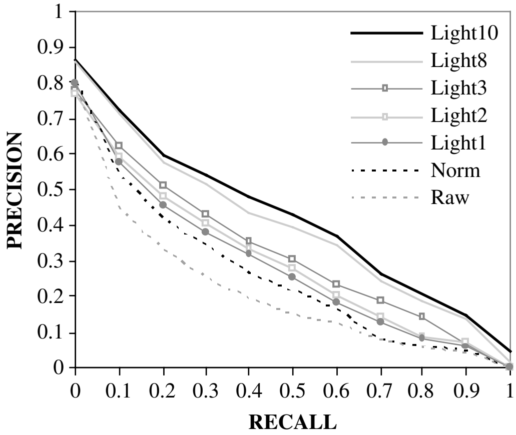

Fig. 12.1. Monolingual 11 point precision for basic stemmers, unexpanded queries

[2] All significance tests were conducted using the Wilcoxon test 0 with a criterion of p<.05 for significance.

Table 12.2. Monolingual average precision for basic stemmers, unexpanded queries

Stemmer	Average Precision	Percent Change
raw	.196	
norm	.241	+22.9
light1	.273	+39.3
light2	.291	+48.3
light3	.317	+61.8
light8	.390	+98.7
light10	.413	+100.1

12.2.4 Comparison with Morphological Analysis

The Khoja stemmer described in 12.1.3 was used to find roots for indexing and retrieval. Average precision for the Khoja stemmer is .341, significantly worse than light10 (p<.01). A comparison of this approach with a raw, normalized, and light2 and light10 stemmers can be seen in Figure 12.2. A similar experiment with query expansion showed similar results, seen in Figure 12.3. In Figure 12.3, *Raw* is the original *raw* condition with unexpanded queries, and *RawExp* refers to the raw condition with query expansion. As expected, average precision is higher with expanded queries, but the same pattern of results holds. In particular, the light10 stemmer is significantly more effective than the khoja stemmer.

12.2.5 Cross-language Retrieval

The Khoja morphological analyzer was also compared with the stemmers in a cross-language retrieval experiment, for generality. The cross-language experiments

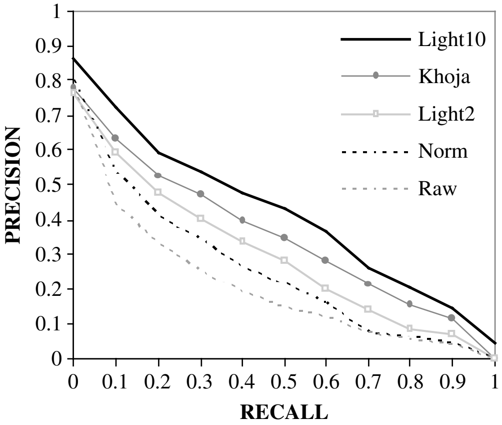

Fig. 12.2. Khoja morphological analyzer versus light stemming, unexpanded queries

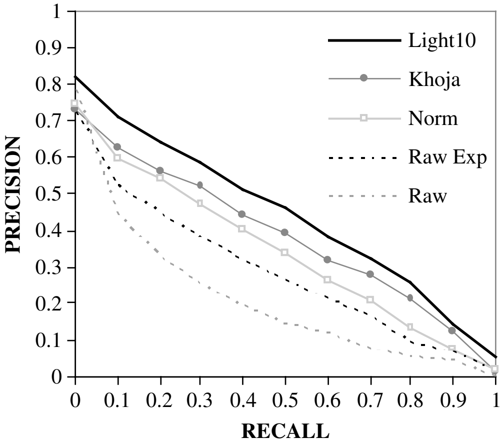

Fig. 12.3. Khoja morphological analyzer versus light stemming, expanded queries

reported here were carried out using the 25 English TREC-2001 queries and the same Arabic AFP_ARB corpus used for the monolingual experiments. Our approach was the common dictionary-based approach, in which each English query word was looked up in a bilingual dictionary. All the Arabic translations for that word were gathered inside an INQUERY #syn (synonym) operator. For an Arabic-English dictionary, we used a lexicon collected from several online English-Arabic and Arabic-English resources on the web, described more completely in [38]. Query expansion was carried out in conjunction with stemming. When English queries were expanded, 5 terms were added from the top-10 documents. When Arabic queries were expanded, 50 terms were added from the top-10 documents, as described in [38].

Figure 12.4 shows precision on unexpanded queries for cross-language retrieval at 11 recall points for raw, norm (normalization and stop word removal), light10 (light10 stemming with stop word removal), and khoja stemmers. Figure 12.5 shows the same information for retrieval with query expansion.

Table 12.3 shows uninterpolated average precision for unexpanded and expanded queries.

The cross-language results are somewhat different from the monolingual results in comparing the light stemmers with the Khoja morphological analyzer. Raw retrieval without any normalization or stemming is far worse for cross-language retrieval than for monolingual retrieval. This is probably because many of the Arabic words occurred in vocalized form (with diacritics) in the online dictionary we used for cross-language retrieval. Without normalization these dictionary entries do not match their counterparts in the corpus. Other differences from the monolingual case are that the *light10* stemmer is far better than the root stemmer, *khoja*, which is no better than normalization for cross-language retrieval. For

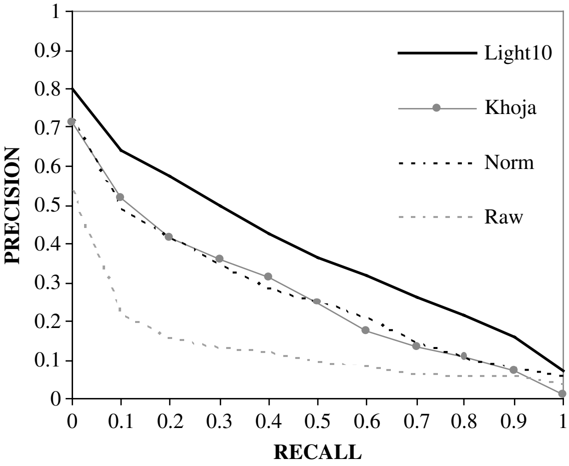

Fig. 12.4. Cross-Language 11 point precision for unexpanded queries

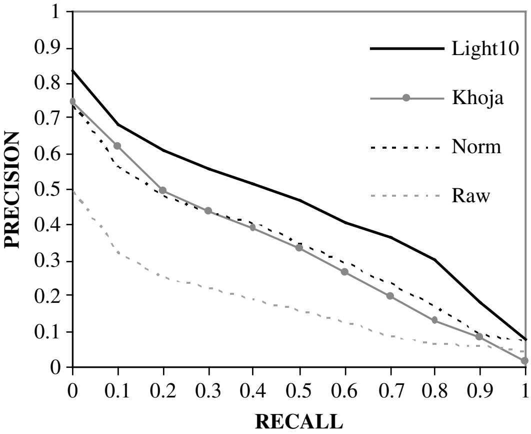

Fig. 12.5. Cross-language 11 point precision for expanded queries

Table 12.3. Cross-language average precision different stemmers, unexpanded and expanded queries

Stemmer	Raw	Norm	Khoja	Light10
Average Precision	.113	.262	.260	.384
Percent Change		+133	+130	+240
With English Query Expansion				
Average Precision	.139	.306	.308	.425
Percent Change		+120	+121	+206
With English and Arabic Query Expansion				
Average Precision	.163	.336	.321	.447
Percent Change		+106	+97	+174

cross-lingual retrieval, roots are probably even less appropriate as look-up units than they are for monolingual retrieval. In cross-lingual retrieval based on dictionary look-up, if we look up the root for each query word, we get far too many translations, and most of them are incorrect.

12.2.6 Discussion

Although stemming is difficult in a language with complex morphology like Arabic, it is particularly important. For monolingual retrieval, we saw around 100% increase in average precision from raw retrieval to the best stemmer. The best stemmer in our experiments, light10, was very simple and did not try to find roots or take into account most of Arabic morphology. It is probably not essential for the stemmer to yield the correct forms, whether stems or roots. It is sufficient for it to group together most of the forms that belong together.

12.3 New Studies of Stemming via Morphological Analysis

Since 2002, more morphological analysis tools have become available. It is also clear that there are probably better ways to use morphological analysis in stemming than simply to use the roots for indexing. In this part of the chapter, we report research on using the Buckwalter morphological analyzer, and the Diab tokenizer and part of speech tagger to aid the stemming process.

12.3.1 Buckwalter Morphological Analyzer

Tim Buckwalter's morphological analyzer has been made available through the Linguistic Data Consortium (LDC) [39]. It takes as input Arabic words with or without short vowels and performs morphological analysis and POS tagging using three dictionaries and three compatibility tables. The three dictionaries list possible prefixes, Arabic stems, and possible suffixes. The three compatibility tables indicate (1) compatible prefix/stem category pairs, (2) compatible prefix/suffix category pairs, and (3) compatible stem/suffix category pairs. The analyzer performs tokenization, word segmentation, dictionary lookup, compatibility checks, and lists all the possible analyses of each word. For example, for the word الشمالية (AlšmAlyħ), we get the following output according to Buckwalter's transliteration:

```
INPUT STRING: الشمالية
LOOK-UP WORD: Al$mAlyp
  SOLUTION 1: (Al$amAliy~ap) [$amAliy~_1] [$amAliy~]Al/DET+$amAliy~/ADJ+ap/NSUFF_FEM_SG
        (GLOSS): the + north/northern + [fem.sg.]
  SOLUTION 2: (Al$imAliy~ap) [$imAliy~_1] [$imAliy~] Al/DET+$imAliy~/ADJ+ap/NSUFF_FEM_SG
         (GLOSS): the + leftist + [fem.sg.]
```

Note that the second field in square brackets in each solution line is one we added to the morphological analyzer program, AraMorph.pl, to give the stem in a form that is more useful to us. The example illustrates some interesting properties of the analyzer. Although much of our corpus does not include short vowels, the analyses have short vowels. In fact, the stems yielded by the two different solutions above differ only in the short vowels.

It is straightforward to use this morphological analysis for stemming, because it analyzes tokens into up to three parts: prefix, stem, and suffix. To stem we simply remove all prefixes and suffix and use the remaining stems, normalized to be comparable to our light stemmers. A potential problem is in dealing with multiple different analyses for the same word. However, once short vowels were removed and other normalization was performed, many of the different analyses actually yielded the same stem. In the example above, the two different stems, šamAliy~ and šimAliy~, both become the same stem, šmAly, after normalization. Ultimately, the vast majority of words had unique stems. In one sample of 18,035 words, 14,878 (82%) were found to have exactly one solution, 2322 (16%) had more than one solution, and 829 (less than 1%) had no solution.

In particular, the following steps were performed on each file of the AFP_ARB corpus:

1. Run the modified version of AraMorph.pl, to find all the analyses for each word
2. Normalize each stem in each solution by removing short vowels, converting all forms of *alif* to bare *alif*, and changing *alif maqS ra* (ى) to *y ’* (ي).
3. If there is exactly one normalized stem, replace the word with the stem. If there were no solutions found, or more than one distinct normalized stem, use the normalized form of the original word.

To address the problem of what to do if the morphological analyzer gave more than one possible stem, Xu, Fraser, and Weischedel [57] implemented a system that used both analyses when a word had more than one solution, but found the results were not significantly different from one that left words with multiple analyses unstemmed. We decided to implement a second version of a Buckwalter-based stemmer (*Buckwalter+*) that applied light10 to words if the analyzer found zero analyses or more than one analysis.

12.3.2 Diab Tokenizer, Lemmatizer and POS Tagger

The Diab morphological analysis tools are available for download on the internet. Their distribution ArabicSVMTools [19] includes the models they trained, which we used without training our own models. Their tokenization segments clitics (prepositions, conjunctions, and some pronouns) from stems, and the part of speech tagger labels each segment with one of 24 parts of speech from a tag set collapsed from the 135 tags created by Buckwalter’s AraMorph. We noted, first of all, that the tokenization part of the process separates some of the same segments that a stemmer should remove – it separates و (w) from the beginnings of words, and determiners like ال (Al). It also separates some prepositions like ب (b) and ل (l), which light10 does not do, unless they precede ال (Al). It also separates some suffixes, like possessive pronoun enclitics. The POS tagger then tags these segments with a POS label, which lets us identify closed-class segments and remove them to accomplish stopword removal. It also allows us to remove additional suffixes contingent on part of speech.

To use the tagger for stemming, we first modified our query and corpus files to contain one sentence per line, because the analyzer operates on sentences. We then ran the tokenizer, lemmatizer, and POS tagger on the sentences, We removed segments with the following tags: CC, DT, RP, PRP, PRP$, CD, IN, WP, WRB, PUNC, NUMERIC_COMMA (conjunction, determiner, particle, personal pronoun, possessive personal pronoun, cardinal number, subordinating conjunction or preposition, relative pronoun, wh-adverb, punctuation). This amounts to a much weaker stemmer than light10, because it removes almost no suffixes. Therefore we tested three other stemmers derived from this morphological analyzer. Our goal was to remove possible plural and dual endings only from words identified as plural nouns and adjectives. Unfortunately, while singular and plural nouns (and singular and plural proper nouns) received distinct tags, adjectives all received the same tag, JJ, so we could not easily determine which were plural or dual. Therefore, we tried two versions of plural suffix removal, described below as Diab2 and Diab3. In an analogue to the Buckwalter+ condition, in which we blindly performed light10 stemming if a word did not yield a unique stem, we also have a Diab+ condition, in which we remove light10 suffixes regardless of part of speech.

To summarize the 4 stemmers derived from the morphological analysis tools:

Diab: tokenization, morphological analysis, remove closed class segments
Diab+: Diab, then remove light10 suffixes
Diab2: Diab, then remove possible plural and dual endings (At, wn, yn, w, An, y) from segments marked as plural nouns or plural proper nouns
Diab3: Diab, then remove (At, wn, yn, w, An, y, yħ, ħ) from any segments marked as nouns or adjectives.

12.3.3 Comparison of New Morphological Stemmers with Light Stemmer

These experiments were carried out in much the same way as those in Section 12.2.3, except for a larger query set. In addition to the 25 queries from TREC2001, there were 50 queries from TREC2002 for a total of 75. The same Arabic corpus was used for retrieval. Figure 12.6 and Figure 12.7 show monolingual retrieval for the 75 queries.

Table 12.4 shows average precision for all the stemming conditions tested, with and without query expansion. The boldface type indicates that the average precision was significantly worse than the corresponding light10 condition.

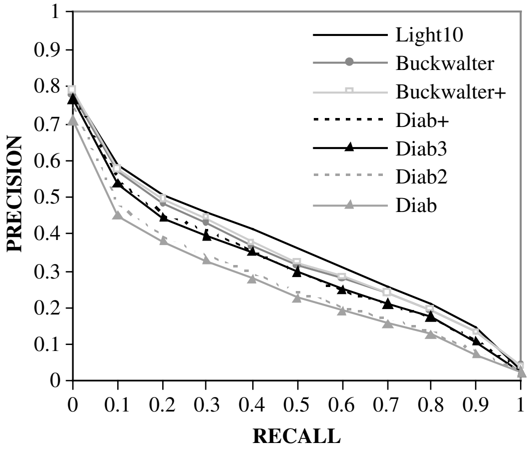

Fig. 12.6. Monolingual 11 point precision for 75 unexpanded queries

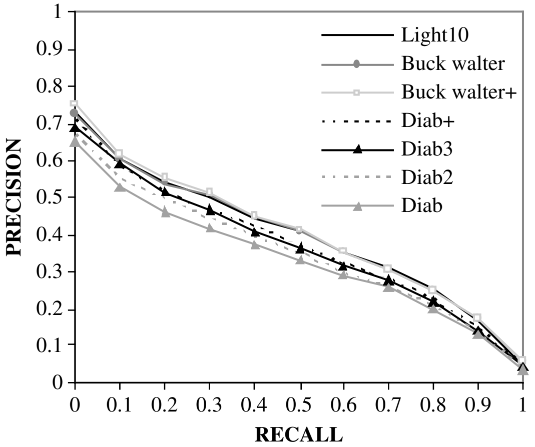

Fig. 12.7. Monolingual 11 point precision for 75 expanded queries

Table 12.4. Average Precision for 75 expanded queries, comparison of morphological stemmers with light10

Stemmer	Unexpanded	Expanded
Light10	.353	.387
Buckwalter	.330	.386
Buckwalter+	.334	.390
Diab	.247	.322
Diab2	.257	.336
Diab3	.302	.354
Diab+	.302	.356

Without query expansion, all of the morphological analysis conditions are significantly worse than light10. With query expansion, the Buckwalter stemmers equal the performance of light10. For both expanded and unexpanded queries, all the Diab stemmers are significantly worse than the Buckwalter stemmers. Diab+ and Diab3 were almost identical, that is, blindly removing the set of light10 suffixes from all words gives the same performance as removing suffixes from nouns and adjectives. Diab+ and Diab3 were significantly better than Diab2, which is significantly better than Diab.

The poor performance of the basic Diab stemmer is not surprising - it is performing stemming that is most comparable to light3 in Section 12.2.3. But we expected it to be more improved by POS-dependent suffix removal, and by its more complete stop word removal. An examination of the query set and a sample AFP article after tokenization and POS tagging showed more tokenization mistakes than we expected. For example, the tokenizer sometimes did not separate the definite article ال (Al).

The queries were made up of a non-sentence (title) line, followed by one to three sentences of description, as in Table 12.5 and Table 12.6. Because the tokenizer was trained on complete sentences, it did not work well on titles. It often failed to segment *Al* when the first word of a title was a noun, as in Table 12.5. Table 12.6 shows an example of incorrect segmentation of *b* from the front of a word in a title, but not in the complete sentence (the description).

Table 12.5. Example of Tokenization, TREC 2002 query AR30

	Title	Description
English	Iraqi satellite television	What is the importance of satellite television in Iraq?
Arabic	**التلفزيون الفضائي في العراق**	**ما اهمية التلفزيون الفضائي في العراق؟**
Transliterated	Altlfzywn AlfDAŷy fy AlçrAq	mA Ahmyħ Altlfzywn AlfDAŷy fy AlçrAq?
Tokenized	Altlfzywn Al fDAŷy fy Al çrAq	mA Ahmyħ Al tlfzywn Al fDAŷy fy Al çrAq?

Table 12.6. Example of Tokenization, TREC2002 query AR32

	Title	Description
English	Caspian Beluga Conservation	What Beluga conservation projects are present in the Caspian region?
Arabic	**صيانة بلوغا في بحرقزوين**	**ما هي المشاريع لصيانة البلوغا في بحر قزوين؟**
Transliterated	SyAnħ blwγA fy bHrqzwyn	mA hy Al mšAryς l SyAnħ Al blwγA fy bHr qzwyn
Tokenized	SyAnħ b lwγA fy bHrqzwyn	mA hy Al mšAryς l SyAnħ Al blwγA fy bHr qzwyn

Performance was hurt by these tokenization errors. Note in the two examples that the tokenization errors were serious - they occurred in important content words in the query, but the same words were correctly tokenized in the description part of the query. Performance would have been hurt even more on title-only queries.

12.4. Conclusions

Stemming has a large effect on Arabic information retrieval, far larger than the effect found for most other languages. For monolingual retrieval we have demonstrated improvements of around 100% in average precision due to stemming and related processes, and an even larger effect for dictionary-based cross-language retrieval. This stemming effect is very large, compared to that found in many other stemming studies, but is consistent with the hypothesis of Popovič and Willett [49] and Pirkola [48] that stemming should be particularly effective for languages with more complex morphology.

The best stemmer was a light stemmer that removed stop words, definite articles, and و ("and") from the beginning of words, and a small number of suffixes from the end of words (*light10*). With query expansion, *light10* yielded results comparable to that of the top performers at TREC, monolingual and cross-language. We have now compared light stemming with several different stemming approaches based on morphological analysis: indexing roots returned by morphological analysis, indexing stems returned by morphological analysis, and somewhat more intelligent stemming based on part-of-speech assignments. Although we can equal the light stemmer with stemming based on Buckwalter's morphological analysis, we have not been able to attain significantly better performance using morphological analysis.

Given the morphological complexity of Arabic, why would a morphological analyzer not perform better than such a simple stemmer? We hypothesize several factors. First, morphological analyzers make mistakes, particularly on names. In dictionary-driven Buckwalter approach, words are exempted from stemming if they are not found in the lexicon. With the Diab approach, we observed many mistakes of tokenization in morphological analysis, which prevented words from getting the correct part of speech, and therefore did not undergo the correct POS-dependent modifications. If one took a sample of Arabic text with complete sentences, the tokenization and POS tagging would have fewer errors. Note, however, that Arabic text contains so many definite articles that one could obtain the claimed >99% tokenization accuracy simply by removing *Al* from the beginnings of words.

Second, models used in IR treat documents and queries as "bags of words," or at best, bags of unigrams, bigrams, and trigrams. Our current retrieval models may not be able to use the information provided by morphological analysis.

Third, light stemming is robust. It does not require complete sentences. It does not try to handle every single case. It is sufficient for information retrieval that many of the most frequently occurring forms of a word be conflated. If an occasional form is missed, it is likely that other forms of the same word occur with it in the same documents, so the documents are likely to be retrieved anyway.

Fourth, it is still not clear what the correct level of conflation should be for IR. Clearly, we do not want to represent Arabic words by their roots, and equate all words derived from the same root. But we still believe that light stemmers are too weak. None of the light stemmers correctly groups broken plurals with their singular forms.

These studies are only a beginning. We have not ruled out the possibility that a better morphological analyzer, and better use of morphological analysis to conflate words, could work better than a light stemmer. We have only tried a few obvious alternatives. Ultimately, one would like to be able to conflate all the inflected forms of a noun together, including broken plurals, and all the conjugations of a verb, which we cannot do today. Clearly, there is room for future work that makes intelligent use of morphological analysis in information retrieval.

Acknowledgments

We would like to thank Shereen Khoja for providing her stemmer, Nicholas J. DuFresne for writing some of the stemming and dictionary code, Fang-fang Feng for helping with dictionary collection over the web, Mohamed Taha Mohamed, Mohamed Elgadi, and Nasreen Abdul-Jaleel for help with the Arabic language, Victor Lavrenko for the use of his vector and language modeling code. This work was supported in part by the Center for Intelligent Information Retrieval and in part by SPAWARSYSCEN-SD grant numbers N66001-99-1-8912 and N66001-02-1-8903. Any opinions, findings and conclusions or recommendations expressed in this material are the authors' and do not necessarily reflect those of the sponsor.

References

[1] Abu-Salem, H., Al-Omari, M., and Evens, M. Stemming methodologies over individual query words for Arabic information retrieval. *JASIS*, 50 (6), pp. 524–529, 1999.

[2] Al-Fedaghi, S. S. and Al-Anzi, F. S. A new algorithm to generate Arabic root-pattern forms. In *Proceedings of the 11th national computer conference*. King Fahd University of Petroleum & Minerals, Dhahran, Saudi Arabia, pp. 391–400, 1989.

[3] Aljlayl, M., Beitzel, S., Jensen, E., Chowdhury, A., Holmes, D., Lee, M., Grossman, D., and Frieder, O. IIT at TREC-10. In *TREC 2001*. Gaithersburg: NIST, pp. 265–275, 2001.

[4] Al-Kharashi, I. and Evens, M. W. Comparing words, stems, and roots as index terms in an Arabic information retrieval system. *JASIS*, 45 (8), pp. 548–560, 1994.

[5] Allan, J. Callan, J. Collins-Thompson, K. Croft, B. Feng, F. Fisher, D. Lafferty, J. Larkey, L. Truong, T. N. Ogilvie, P. Si, L. Strohman, T. Turtle, H. and Zhai, C. The Lemur toolkit for language modeling and information retrieval. http://www.lemurproject.org/˜lemur

[6] Al-Shalabi, R. *Design and implementation of an Arabic morphological system to support natural language processing*. PhD thesis, Computer Science, Illinois Institute of Technology, Chicago, 1996.

[7] Beesley, K. R. Arabic finite-state morphological analysis and generation. In *COLING-96: Proceedings of the 16th international conference on computational linguistics*, vol. 1, pp. 89–94, 1996.

[8] Berlian, V., Vega, S. N., and Bressan, S. Indexing the Indonesian web: Language identification and miscellaneous issues. Presented at Tenth International World Wide Web Conference, Hong Kong, 2001.

[9] Brent, M. R. Speech segmentation and word discovery: A computational perspective. *Trends in Cognitive Science*, 3 (8), pp. 294–301, 1999.

[10] Buckwalter, T. *Qamus: Arabic lexicography*. http://www.qamus.org/

[11] Callan, J. P., Croft, W. B., and Broglio, J. TREC and TIPSTER experiments with INQUERY. *Information Processing and Management*, 31 (3), pp. 327–343, 1995.

[12] Carlberger, J., Dalianis, H., Hassel, M., and Knutsson, O. Improving precision in information retrieval for Swedish using stemming. In *Proceedings of NODALIDA '01–13th Nordic conference on computational linguistics*. Uppsala, Sweden, 2001. http://www.nada.kth.se/~xmartin/papers/-Stemming_NODALIDA01.pdf

[13] Chen, A. and Gey, F. Building an Arabic stemmer for information retrieval. In *TREC 2002*. Gaithersburg: NIST, pp 631–639, 2002.

[14] Darwish, K. Building a shallow morphological analyzer in one day. ACL 2002 Workshop on Computational Approaches to Semitic languages, pp. 47–54, July 11, 2002.

[15] Darwish, K., Doermann, D., Jones, R., Oard, D., and Rautiainen, M. TREC-10 experiments at Maryland: CLIR and video. In *TREC 2001*. Gaithersburg: NIST, pp 549–562, 2001.

[16] Darwish, K. and Oard, D.W. CLIR Experiments at Maryland for TREC-2002: Evidence combination for Arabic-English retrieval. In *TREC 2002*. Gaithersburg: NIST, pp 703–710, 2002.

[17] de Marcken, C. *Unsupervised language acquisition*. PhD thesis, MIT, Cambridge, 1995.

[18] De Roeck, A. N. and Al-Fares, W. A morphologically sensitive clustering algorithm for identifying Arabic roots. In *Proceedings ACL-2000*. Hong Kong, pp 199–206, 2000.

[19] Diab, M. ArabicSVMTools. http://www.stanford.edu/~mdiab/software/ArabicSVMTools.tar.gz. 2004.

[20] Diab, M., Hacioglu, K., and Jurafsky, D. Automatic tagging of Arabic text: From raw test to base phrase chunks. In *Proceedings of HLT-NAACL*, pp 149–152, 2004. http://www.stanford.edu/~mdiab/papers/ArabicChunks.pdf.

[21] Ekmekcioglu, F. C., Lynch, M. F., and Willett, P. Stemming and n-gram matching for term conflation in Turkish texts. *Information Research News*, 7 (1), pp. 2–6, 1996.

[22] Flenner, G. Ein quantitatives Morphsegmentierungssytem fur Spanische Wortformen. In *Computatio linguae II*, U. Klenk, Ed. Stuttgart: Steiner Verlag, pp. 31–62, 1994.

[23] Frakes, W. B. Stemming algorithms. In *Information retrieval: Data structures and algorithms*, W. B. Frakes and R. Baeza-Yates, Eds. Englewood Cliffs, NJ: Prentice Hall, Chapter 8, 1992.

[24] Freund, E. and Willett, P. Online identification of word variants and arbitrary truncation searching using a string similarity measure. *Information Technology: Research and Development*, 1, pp. 177–187, 1982.

[25] Gey, F. C. and Oard, D. W. The TREC-2001 cross-language information retrieval track: Searching Arabic using English, French, or Arabic queries. In *TREC 2001*. Gaithersburg: NIST, pp 16–26, 2002.

[26] Goldsmith, J. Unsupervised learning of the morphology of a natural language. *Computational Linguistics*, 27 (2), pp. 153–198, 2000.

[27] Goldsmith, J., Higgins, D., and Soglasnova, S. Automatic language-specific stemming in information retrieval. In *Cross-language information retrieval and evaluation: Proceedings of the CLEF 2000 workshop*, C. Peters, Ed.: Springer Verlag, pp. 273–283, 2001.

[28] Goweder, A. and De Roeck, A. Assessment of a significant Arabic corpus. Presented at the Arabic NLP Workshop at ACL/EACL 2001, Toulouse, France, 2001. http://www.elsnet.org/arabic2001/goweder.pdf

[29] Greengrass, M., Robertson, A. M., Robyn, S., and Willett, P. Processing morphological variants in searches of Latin text. *Information Research News*, 6 (4), pp. 2–5, 1996.

[30] Hafer, M. A. and Weiss, S. F. Word segmentation by letter successor varieties. *Information Storage and Retrieval*, 10, pp. 371–385, 1974.

[31] Hull, D. A. Stemming algorithms - a case study for detailed evaluation. *JASIS*, 47 (1), pp. 70–84, 1996.

[32] Janssen, A. Segmentierung Franzosischer Wortformen in Morphe ohne Verwendung eines Lexikons. In *Computatio linguae*, U. Klenk, Ed. Stuttgart: Steiner Verlag, pp. 74–95, 1992.

[33] Khoja, S. and Garside, R. *Stemming Arabic text*. Computing Department, Lancaster University, Lancaster, 1999. http://www.comp.lancs.ac.uk/computing/users/khoja/stemmer.ps

[34] Klenk, U. Verfahren morphologischer Segmentierung und die Wortstruktur im Spanischen. In *Computatio Linguae*, *Aufsätze zur algorithmischen und quantitativen Analyse der Sprache*, U. Klenk, Ed. Stuttgart: Steiner Verlag, pp 110–124, 1992.

[35] Kraaij, W. and Pohlmann, R. Viewing stemming as recall enhancement. In *Proceedings of ACM SIGIR96*. pp. 40–48, 1996.

[36] Krovetz, R. Viewing morphology as an inference process. In *Proceedings of ACM SIGIR93*, pp. 191–203, 1993.

[37] Larkey, Leah S., Ballesteros, L., and Connell, M. (2002) Improving stemming for Arabic information retrieval: Light stemming and co-occurrence analysis In *Proceedings of the 25th annual international conference on research and development in information retrieval (SIGIR 2002)*, Tampere, Finland, August 11–15, 2002, pp. 275–282.

[38] Larkey, L. S. and Connell, M. E. Arabic information retrieval at UMass in TREC-10. In *TREC 2001*. Gaithersburg: NIST, 2001.

[39] LDC, Linguistic Data Consortium. Buckwalter Morphological Analyzer Version 1.0, LDC2002L49, 2002. http://www.ldc.upenn.edu/Catalog/.

[40] LDC, Linguistic Data Consortium. Arabic Penn TreeBank 1, v2.0. LDC2003T06, 2003. http://www.ldc.upenn.edu/Catalog/

[41] Lovins, J. B. Development of a stemming algorithm. *Mechanical Translation and Computational Linguistics*, 11, pp. 22–31, 1968.

[42] Mayfield, J., McNamee, P., Costello, C., Piatko, C., and Banerjee, A. JHU/APL at TREC 2001: Experiments in filtering and in Arabic, video, and web retrieval. In *TREC 2001*. Gaithersburg: NIST, pp 332–341, 2001.

[43] McNamee, P., Mayfield, J., and Piatko, C. A language-independent approach to European text retrieval. In *Cross-language information retrieval and evaluation: Proceedings of the CLEF 2000 workshop*, C. Peters, Ed.: Springer Verlag, pp. 129–139, 2000.

[44] Monz, C. and de Rijke, M. Shallow morphological analysis in monolingual information retrieval for German and Italian. In *Cross-language information retrieval and evaluation: Proceedings of the CLEF 2001 workshop*, C. Peters, Ed.: Springer Verlag, 2001. http://staff.science.uva.nl/~christof/Papers/clef-2001-post.pdf

[45] Moulinier, I., McCulloh, A., and Lund, E. West group at CLEF 2000: Non-English monolingual retrieval. In *Cross-language information retrieval and evaluation: Proceedings of the CLEF 2000 workshop*, C. Peters, Ed.: Springer Verlag, pp. 176–187, 2001.

[46] Oard, D. W., Levow, G. -A., and Cabezas, C. I. CLEF experiments at Maryland: Statistical stemming and backoff translation. In *Cross-language information retrieval and evaluation: Proceedings of the CLEF 2000 workshop*, C. Peters, Ed.: Springer Verlag, pp. 176–187, 2001.

[47] NIST. Topic Detection and Tracking Resources. http://www.nist.gov/speech/tests/tdt/resources.htm. Created 2000, updated 2002.

[48] Pirkola, A. Morphological typology of languages for IR. *Journal of Documentation*, 57 (3), pp. 330–348, 2001.

[49] Popovic, M. and Willett, P. The effectiveness of stemming for natural-language access to Slovene textual data. *JASIS*, 43 (5), pp. 384–390, 1992.

[50] Porter, M. F. An algorithm for suffix stripping. *Program*, 14 (3), pp. 130–137, 1980.

[51] Rogati, M., McCarley, S., and Yang, Y. Unsupervised learning of Arabic stemming using a parallel corpus. In *Proceedings ACL-2003,* Sapporo, Japan, pp. 391–398, July 2003. http://acl.ldc.upenn.edu/acl2003/main/pdf/Rogati.pdf

[52] Siegel, S. *Nonparametric statistics for the behavioral sciences*. New York: McGraw-Hill, 1956.

[53] Taghva, K., Elkoury, R., and Coombs, J. Arabic Stemming without a root dictionary. 2005. www.isri.unlv.edu/publications/isripub/Taghva2005b.pdf

[54] Tai, S. Y., Ong, C. S., and Abdullah, N. A. On designing an automated Malaysian stemmer for the Malay language. (poster). In *Proceedings of the fifth international workshop on information retrieval with Asian languages,* Hong Kong, pp. 207–208, 2000.

[55] Xu, J. and Croft, W. B. Corpus-based stemming using co-occurrence of word variants. *ACM Transactions on Information Systems*, 16 (1), pp. 61–81, 1998.

[56] Xu, J., Fraser, A., and Weischedel, R. TREC 2001 cross-lingual retrieval at BBN. In *TREC 2001*. Gaithersburg: NIST, pp. 68–78, 2001.

[57] Xu, J., Fraser, A., and Weischedel, R. Empirical studies in strategies for Arabic retrieval. In *Sigir 2002*. Tampere, Finland: ACM, pp. 269–274, 2002.