Finding Variants of Out-of-Vocabulary Words in Arabic

Abdusalam F.A. Nwesri   S.M.M. Tahaghoghi   Falk Scholer
School of Computer Science and Information Technology
RMIT University, GPO Box 2476V, Melbourne 3001, Australia
{nwesri,saied,fscholer}@cs.rmit.edu.au

Abstract

Transliteration of a word into another language often leads to multiple spellings. Unless an information retrieval system recognises different forms of transliterated words, a significant number of documents will be missed when users specify only one spelling variant. Using two different datasets, we evaluate several approaches to finding variants of foreign words in Arabic, and show that the longest common subsequence (LCS) technique is the best overall.

1 Introduction

The pronunciation of a word in one language is converted into the phonemes of another language through transliteration. This is particularly common with proper nouns. However, phonetics differ across languages, and transliteration usually results in many spellings for the same word. This is an issue even across languages that use substantially the same character set; simple examples would be "colour" and "color" across British and American usage, and "ambience" and "ambiance" across French and English.

A change in character sets compounds the problem: for instance, there are at least 32 English forms for the Arabic name of the Libyan leader "Kaddafi",[1] and Nwesri et al. (2006) have identified 28 different spellings for the name of the former Serbian president Milosevic in the eleventh Text REtrieval Conference (TREC) Arabic newswire collection. Users typically submit only one spelling variant in their query, and current Arabic text retrieval systems return only documents that contain that variant (Abdelali et al., 2004). We apply techniques used to identify similar strings in other languages such as English, and present a novel approach to identify and retrieve different variants of foreign words in Arabic.

2 The Arabic Language

Arabic is a Semitic language written from right to left, with most words derived from three-character root words. The Arabic alphabet has 28 characters, each with a distinct sound. Short vowels do not have any associated characters, but are instead indicated by diacritics attached to other characters. For example, the letter ف /f/ with the diacritic Fatha فَ is pronounced /fa/,[2] with the diacritic Kasra فِ is pronounced /fɪ/, and with the diacritic Damma فُ is pronounced /fʊ/.

In general written Arabic, diacritics are not indicated; readers must rely on context to determine implicit diacritics, and so how the word should be pronounced. For example, some of the variants of the word كتب are كَتَبَ /kataba/ ⟨he wrote⟩, كُتُب /kʊtʊb/ ⟨books⟩, or كُتِبَ /kʊtɪba/ ⟨is written⟩.

There are also three long vowels — represented by the letters {ا ي و} — that are more pronounced than short vowels. For instance, the letter ف can be followed by the long vowel ا /aː/ to form فا /faː/, by و /uː/ to form فو /fuː/, and by ي /iː/ to form في /fiː/.

2.1 Foreign Words

From an information retrieval (IR) perspective, foreign words in Arabic can be classified into two general categories: translated and transliterated (Nwesri et al., 2006). Translated words, sometimes referred to as Arabised words, are foreign words that are modified or remodelled to conform to Arabic word

[1] http://www.geocities.com/Athens/8744/spelling.htm

[2] We use the International Phonetic Alphabet.

Proceedings of the 5th Workshop on Important Unresolved Matters, pages 49–56,
Prague, Czech Republic, June 2007.

paradigms, and are well assimilated into the language. The assimilation process includes changes in the structure of the borrowed word, including segmental and vowel changes, addition or deletion of syllables, and modification of stress patterns (Al-Qinal, 2002). Foreign words of this category usually have a single consistent spelling variant, for example فيروس ⟨virus⟩, ارشيف ⟨archive⟩, and راديو ⟨radio⟩.

Where equivalent native terms are not available early enough for widespread adoption, foreign terms are used directly with their original pronunciation represented using Arabic letters. As these do not appear in standard Arabic lexicons — that may include adopted words — they are considered to be Out-Of-Vocabulary (OOV) words.

With transliterated words, the phonemes of a foreign word are replaced with their nearest Arabic equivalents. Since Arabic phonemes cannot represent all phonemes found in other languages, the original phonemes are usually not represented uniformly by different transliterators, resulting in multiple spelling variants for the same foreign word (Stalls and Knight, 1998).

Faced with the need to use new foreign terms, native speakers often cannot wait for formal equivalents to be defined. This is particularly true for news agencies, which encounter new foreign nouns and technical terms daily. This urgency leads to more transliteration than translation, with the associated problem of multiple spellings.

2.2 Spelling Variants

In Arabic, short vowels must be indicated using diacritics, but these are rarely used in general text, and there are no standard rules on when and where diacritics must be indicated. Context does not help in predicting diacritics for foreign words such as proper nouns or technical terms, and so long vowels are often used to make the pronunciation explicit in the spelling of the word without relying on diacritics. This, too, is subject to variation; some transliterators add a long vowel after each consonant in the word, while others add just enough long vowels to clarify word segments with ambiguous pronunciation.

The absence of certain sounds in Arabic, and varying pronunciations across dialects, also contributes to the multiplicity of spellings. For instance, the sound /g/ has no standard equivalent in Arabic, since transliterators represent it according to how they pronounce it. For instance, the English letter G /g/ is at times mapped to the Arabic letters غـ /ɣ/, قـ /q/, or جـ /ʒ/ (Abduljaleel and Larkey, 2003); we have also observed it mapped to the letter كـ /k/: غورباتشوف, قورباتشوف, جورباتشوف, and كورباتشوف are transliterations of ⟨Gorbachev⟩ we have found on the Web.

Similarly, the interpretation of character combinations varies between transliterators. Moreover, Typographical and phonetic errors during transliteration may add even more variants (Borgman and Siegfried, 1992).

2.3 Retrieval of Variants

When different variants of a word exist, only a subset of related documents can be found when the search uses only one variant. Typical search engine users are unlikely to recognise the problem and hence do not add other variants to their query. Currently, major search engines such as Google, Yahoo, and MSN search use exact match for Arabic search, and no publicly available Arabic Information Retrieval (AIR) system has been reported to retrieve different spelling variants (Abdelali et al., 2004).

In this paper we explore how the different variants of a foreign word may be captured. We test existing similarity techniques, and introduce three techniques to search for variants of foreign words in Arabic. In the first technique, we convert different variants to a single normalised form by removing vowels and conflating homophones. In the second technique, we extend the well-known Soundex technique — commonly used to identify variants of names in English — to the OOV problem in Arabic, and in the third technique, we modify the English Editex algorithm to identify similar foreign words in Arabic.

3 Related Work

Approaches to identify similar-sounding but differently spelt words have been heavily investigated in English; among these are techniques that make use of string or phonetic similarity.

String similarity approaches include the Edit Distance (Hall and Dowling, 1980), used to measure the similarity of two strings by counting the minimal number of character insertions, deletions, or replacements needed to transform one string into another. To transpose a string s of length n into a string t of length m, $edit(m, n)$ computes the minimal steps required as follows:

$$\begin{aligned}
&edit(0, 0) = 0\\
&edit(i, 0) = i\\
&edit(0, j) = j\\
&edit(i, j) = min[edit(i-1, j) + 1,\\
&\qquad\qquad edit(i, j-1) + 1,\\
&\qquad\qquad edit(i-1, j-1) + d(s_i, t_i)]
\end{aligned}$$

where $d(s_i, t_i) = 1$ if $s_i = t_i$, 0 otherwise.

This measure can be used to rank words in the collection with respect to a query word. Zobel and Dart (1995) showed that Edit Distance performed the best among the techniques they evaluated for matching English names. It is not known how this technique will perform with Arabic words.

Another candidate approach that can be used to identify similar foreign words in Arabic is n-grams (Hall and Dowling, 1980). This approach is language independent; the strings are divided into grams (substrings) of length n, and the similarity of the strings is computed on the basis of the similarity of their n-grams. Pfeifer et al. (1996) compute the similarity as the number of shared grams divided by the total number of distinct grams in the two strings.

$$gramCount = \frac{\mid G_s \cap G_t \mid}{\mid G_s \cup G_t \mid}$$

where G_s is the set of grams in string s. For example, with n=2, the similarity of "ahmed" and "ahmmed" using this measure is 0.8 because both strings contain the four 2-grams `ah`, `hm`, `me`, and `ed`, while there are five distinct 2-grams across the two strings.

Gram distance (Ukkonen, 1992) is another string similarity technique. When grams are not repeated – which is the case in names — the similarity is computed as (Zobel and Dart, 1996):

$$gramDist(s, t) = \mid G_s \mid + \mid G_t \mid - 2 \mid G_s \cap G_t \mid$$

According to this measure, the similarity between "ahmed" and "ahmmed" is 1.

With the Dice (1945) measure, the similarity of strings s and t is computed as twice the number of common n-grams between s and t, divided by the total number of n-grams in the two strings:

$$Dice(s, t) = \frac{2\times \mid G_s \cap G_t \mid}{\mid G_s \mid + \mid G_t \mid}$$

where G_s denotes the set of n-grams in s, and G_t denotes the set of n-grams in t.

The longest common subsequence (LCS) algorithm measures the similarity between two strings based on the common characters in the two strings (Wagner and Fischer, 1974; Stephen, 1992). Similarity is normalised by dividing the length of the common subsequence by the length of the longer string (Melamed, 1995). The similarity between between "ahmed" and "ahmmed" is (5/6=0.833).

Phonetic approaches to determine similarity between two words include the well-known Soundex algorithm developed by Odell and Russell, patented in 1918 and 1922 (Hall and Dowling, 1980). This has predefined codes for the sounds in a language, with similar-sounding letters grouped under one code. During comparisons, all letters in a word bar the first one are encoded, and the resulting representation is truncated to be at most four characters long. A variant of Soundex is the Phonix algorithm (Gadd, 1990), which transforms letter groups to letters and then to codes; the actual mappings are different from Soundex. Both Soundex and Phonix have been reported to have poorer precision in identifying variants of English names than both Edit Distance and n-grams (Zobel and Dart, 1995).

Aqeel et al. (2006) propose an Arabic version of English Soundex (Asoundex-final). They include diacritics in a list of Arabic names, and created queries by altering some of these names by adding, deleting, or inserting characters.

Most Arabic names are meaningful words — for example, محمد ⟨the praised one⟩ — and rarely do have spelling variants. This leads to morphological ambiguity as names may match verbs, pronouns and other categories of the language. We have found that using Asoundex-final with the misspelt query تهصين on an Arabic collection with 35 949 unique words returns تحجيم ⟨exaggeration⟩, تحزن ⟨she becomes sad⟩, تحسم ⟨she resolves⟩, تحسن ⟨she helps⟩, تحسين ⟨improvement⟩, تحصين ⟨immunisation⟩, تحكم ⟨she governs⟩, تهزم ⟨she defeats⟩. Moreover, it is not clear when and how diacritics are removed, nor where the long vowel ي belongs in their implementation.

Editex, developed by Zobel and Dart (1996), enhances the Edit Distance technique by incorporating the letter-grouping strategy used by Soundex and Phonix, and has been shown to have better performance than these two algorithms, as well as Edit Distance, on a collection of 30 000 distinct English names. The similarity between two strings s and t is computed as:

$$edit(0,0) = 0$$
$$edit(i,0) = edit(i-1,0) + d(s_i - 1, s_1)$$
$$edit(0.j) = edit(0, j-1) + d(t_j - 1, t_j)$$
$$\begin{aligned} edit(i.j) = min[&edit(i-1,j) + d(s_i - 1, s_i), \\ &edit(i, j-1) + d(t_j - 1, t_j), \\ &edit(i-1, j-1) + r(s_i, t_j)] \end{aligned}$$

where: $r(s_i, t_j)$ is 0 if s_i=t_j, 1 if $group(s_i)$=$group(t_j)$, and 2 otherwise; and $d(s_i, t_j)$ is 1 if $s_i \neq t_j$ and s_i is "h" or "w", and $r(s_i, t_j)$ otherwise.

4 Data

We used two different data sets. The first set is generated from text crawled from the Web, and the second is prepared by manual transliteration of foreign words from English to Arabic.

4.1 Crawled Data

This set is derived from a one-gigabyte crawl of Arabic web pages from twelve different online news sites. From this data we extracted 18 873 073 Arabic words, 383 649 of them unique. We used the Microsoft Office 2003 Arabic spellchecker to build a reference list of OOV words. To avoid duplicates in the 40 514 OOV words returned by the spellchecker, we removed the first character if it is an Arabic preposition, and if the string remaining after that character exists in the collectionWe also removed the definite article "Al" to obtain a list of 32 583 words. Through manual inspection, we identified 2 039 foreign words.

To evaluate alternative techniques, we use a reference list of foreign words and their variants. To identify variants, we generated all possible spelling variants of each word according to the patterns we describe in Section 4.1.1, and kept only the patterns that exist in our collection; 556 clusters of foreign words remain.

Table 1: Variants of the word "Beckham" generated by adding vowels

بكم	باكم	بوكم	بيكم
بكوم	باكوم	بوكوم	بيكوم
بكام	باكام	بوكام	بيكام
بكيم	باكيم	بوكيم	بيكيم

4.1.1 Generation of Variants

To generate foreing words variants, we first remove any vowels and then reinsert vowel combinations of the three long vowels {ا ي و} between the consonants that remain. For a word of length n, this process generates $4^{(n-1)}$ variants. Consider the word بيكام ⟨Beckham⟩. We remove vowels to obtain بكم, and then add all possible vowels to obtain the variants shown in Table 1.

As discussed in Section 2.2, inconsistent representation of sounds between transliterators adds to the variations in spelling. Thus, the number of possible transliterations for a foreign word is given by $4^{(n-1)}$ multiplied by the number of possible transliterations for each of its consonants. In our example, the letter ق /q/ may also be used in place of ك /k/, and so we generate another set; since the representation tends to be consistent within a given word, we need to create only as many sets as there are Arabic representations for the sound.

We validate the generated variants against our collection and keep only those that appear in the crawled text. For our example word "Beckham", we found only two correct variants: بيكام and بيكم. Some of the generated variants could be correct Arabic words that would be valid when checked against the collection. Many of the generated clusters were found to be noisy – that is, they included many native Arabic words. We manually corrected these clusters by removing unrelated Arabic words. The average cluster length is 2.8 words; the smallest cluster has two variants, and the largest has nine, with a total of 1 718 words.

4.2 Transliterated Data

Our second collection reflects one pattern in which OOV words are introduced by ordinary users transliterating English words into Arabic. We extracted a list of 1 134 foreign words from the TREC 2002 Arabic collection, and passed these to the Google translation engine to obtain their English equivalents. We manually inspected these and corrected any incorrect translations. We also removed the 57 words mapped by Google to multiple English words. These are usually a word and a possible conjunction or preposition. For example the word لوكسمبرج ⟨Luxembourg⟩ is transliterated to ⟨For June⟩. We passed the English list to seven Arabic native speakers and asked them to transliterate each word in the list back into Arabic, even if the word has an Arabic equivalent. Four of the translators are PhD candidates in the sciences or engineering, and have finished an advanced-level English course; the other three are currently enrolled in an intermediate-level English course. Participants were asked to type in their transliteration next to each English word. We noticed that some transliterators had only basic computing skills, and made many spelling mistakes. For example, instead of typing the character ا, we found that transliterators sometimes mistakenly type لـ.

We clustered transliterations by the original English words, removed duplicates from each cluster, and also removed 103 clusters where all transliterators agreed on the same version of transliteration. This left 3 582 words in 207 clusters of size 2, 252 clusters of size 3, 192 clusters of size 4, 149 clusters of size 5, 93 clusters of size 6, and 47 clusters of size 7. Finally, we incorporated these transliterations into a list with 35 949 unique Arabic native words prepared by Nwesri et al. (2006).

5 Algorithms

We propose three algorithms to identify foreign words in Arabic text. The first is normalisation, which aims to handle different types of typographical errors described in Section 2.1. The second and third techniques are extensions to the English Soundex and Editex techniques.

5.1 Normalisation

To deal with different typographical styles in writing foreign words, we first remove vowels from every foreign term. We keep vowels unchanged if they are the first or the last characters of the word, since they are generally pronounced in Arabic. The long vowel letters are sometimes used as consonants, and these may be followed immediately by another long vowel. For example, the vowel letter ي /i/ may be followed by the long vowel و /u:/ to form يو /ju:/. For such cases, we keep the first vowel and remove the second. Two vowels can also be used together as diphthongs, as in او /aw/ and اي /aj/. We retain vowels that are followed by another vowel or preceded by a vowel that forms a diphthong. We also conflate similar consonants based on statistical analysis of letter mappings between English and Arabic (Abduljaleel and Larkey, 2003; Stalls and Knight, 1998), and confirming through a web search that these consonants are used interchangeably in web documents.[3] Table 2 shows all consonants we consider to be equivalent.

Table 2: Normalisation of equivalent consonants to a single form

Original				Normalised
س	ش	ز	ص	س
ت	ط			ط
ج	غ	ك	ق	غ
ث				ت

Our process may lead to ambiguity where a similar native word exists; for instance, the spelling variants بيكام and بيكم for ⟨Beckham⟩ are normalised to بكم, which is identical to the Arabic word meaning either ⟨how much⟩ or ⟨in you⟩. Adding a custom prefix to the normalised form is one way to address this issue; we add the letter "ة" to the beginning of each normalised word. For example, variants for Beckham are thus normalised to ةبكم. Since the letter ة never occurs at the beginning of any Arabic word, no ambiguity remains.

5.2 Phonetic Approach

Our phonetic algorithm aims to replace similar sounds with a single code. As noted earlier, we do not envisage that this algorithm has use for native Arabic words, as these are usually distinct, and pro-

[3] All phonetic groups are created based on transliteration mapping between English and Arabic letters

Table 3: Mappings for our phonetic approach

Characters	Code
ا و ي	0
ة ت ط ث ظ ض	1
س ش ز ص	2
د ذ	3
ج غ ك ق	4
ع ه ح	5
ن	6
م	7
ف	8
ل	9
ب	A
ر	B
خ	C

nunciation is rarely ambiguous. Table 3 shows Arabic letters and their corresponding codes. To normalise foreign words, we replace each letter but the first by its phonetic code, and drop any vowels. We have found — as have (Aqeel et al., 2006) and (Zobel and Dart, 1996) — that it is better to encode all letters, rather than only the first four characters; for brevity, we show only the results for coding all characters, under the label "Soutex".

5.3 Arabic Editex

Based on groups identified in Table 4, we have modified the Editex algorithm of Zobel and Dart (1996). It works as in English except that we drop the functionality used to consider the two silent characters in the English version as silent characters in Arabic are rare and usually occur at the beginning or at the end of the word. Specifically, we replace $d(s_i, t_j)$ by $r(s_i, t_j)$. We call the Arabic version of this algorithm "AEditex".

6 Evaluation

To evaluate the effectiveness of our approaches, we consider each word in the list to be a query, and pose this to the entire collection. The query result should be other words in the same cluster. We consider every word to be a query to avoid any bias towards string similarity techniques as phonetic based

Table 4: AEditex letter groups

Characters	Group
ا و ي	0
ت ث	1
ت ط	2
ظ ض	3
س ش	4
س ص	5
ز س	6
د ذ	7
ج غ ك ق	8

techniques fail to capture misspelled words whereas string similarity techniques do.

The results returned by the different algorithms described in the previous section are not directly comparable, as some algorithms return ranked results and others return unranked results. Ranked results could also form a weak ordering in which multiple results belong to the same rank (Raghavan et al., 1989). Standard information retrieval measures are not appropriate for evaluating such techniques. Zobel and Dart (1996) address this by using standard precision and recall, but randomly permute results of equal ranks and calculate the average of recall and precision over ten different permutations. Raghavan et al (1989) propose a different measure called *Probability of Relevance* (PRR). This measure assumes that the user randomly selects a document from the topmost ranks. At any point the precision is defined as the probability that the random examined document is relevant. Recall is the number of relevant documents that the user has seen so far. If we require NR relevant documents — in our case, words — from a ranked result, we start by looking at the top answer and continue until we get to the $NRth$ relevant word at rank k. The PRR measure is calculated as (Raghavan et al., 1989):

$$PRR = \frac{NR}{NR + j + (i.s)/(r+1)}$$

Where j is the number of non-relevant words found in ranks before k, s is the number of remaining relevant words still to be retrieved in rank k, i is the number of non-relevant words in rank k, and r is the

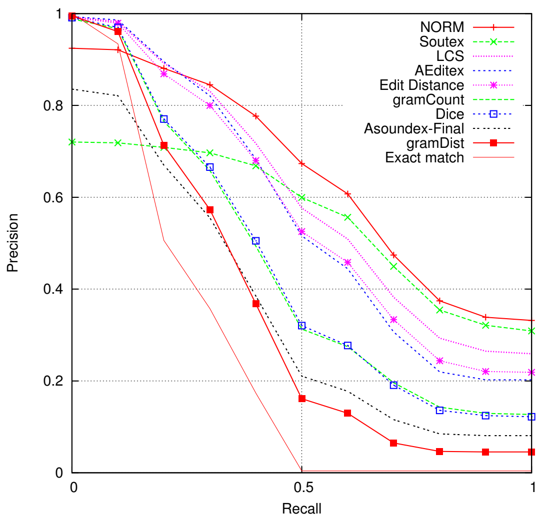

Figure 1: Results on the crawled data

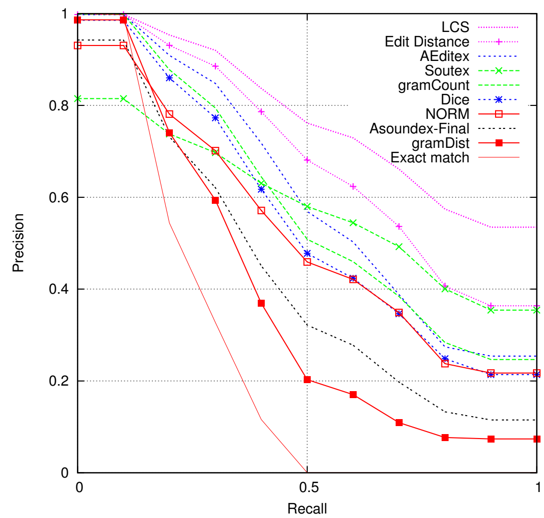

Figure 2: Results on the transliterated data

number of relevant words in rank k. Interpolation is used to smooth results and calculate an average across all queries.

6.1 Results and Discussion

Results from running algorithms using queries in both datasets against their respective collection are shown in Figure 1 and Figure 2. The average precision (average PRR in our case) for each algorithm is shown in Table 5. The algorithms produce significantly better results than exact match (p<0.0001).

On the first data set, NORM performs the best. LCS is the second best algorithm, followed by AEditex and Edit Distance. Soutex shows better performance than all other algorithms except NORM after 50% recall, but performs poorly at lower recall levels. Both the gramCount and Dice algorithms have similar performance with average precision at around 46%. Asoundex-final and gramDist show poorer performance than other algorithms, with average precision at 38%.

Asoundex-final performs poorly; As mentioned earlier, the absence of diacritics in typical Arabic text makes it hard to generalise this technique to retrieve names.

Results from the transliterated dataset generally favour the string similarity algorithms. LCS outperforms all other techniques with an average precision of 78%, followed by Edit Distance at 70%, and then AEditex at 62%. Soutex performs better than both

Table 5: Average precision results

	Data set	
Algorithm	First	Second
NORM	0.660	0.536
LCS	0.619	0.782
Edit Distance	0.572	0.700
AEditex	0.576	0.624
Soutex	0.530	0.590
gramCount	0.451	0.595
Dice	0.457	0.568
Asoundex-final	0.368	0.446
gramDist	0.376	0.401
Exact Match	0.300	0.261

the gramCount and Dice algorithms. It performs better than AEditex at 50% and higher recall levels. NORM performs better than the Asoundex-final and gramDist algorithms. The gramDist algorithm is again the worst. All algorithms showed significant improvements above the baseline (p<0.0001).

Although NORM and Soutex algorithms do not produce the best performance, they have the advantage of being run at index time to encode foreign words which can be later used in retrieval. The alternative algorithms such as Edit Distance are more computationally expensive and can only be used at query time.

7 Conclusion

Foreign words transliterated into Arabic can appear with multiple spellings, hindering effective recall in a text-retrieval system. In this work, we have evaluated nine techniques to find such variants. Edit Distance, Gram Count, Dice, Asoundex-final, Gram Distance, and Longest Common Subsequence are language independent techniques used to find variant names in other languages; Soutex and AEditex are extended techniques to accommodate Arabic Words; and NORM is a novel technique to find OOV variants in Arabic. We show that these techniques are effective for finding foreign word variants. The phonetic approaches generally perform better on a collection of newswire text than on a manually transliterated word list, although our Soutex algorithm performs well on both datasets. LCS was the best of the string-similarity techniques, especially with the manually transliterated dataset, and is the most robust choice overall.

The way the transliterated dataset was created affected the results of phonetic approaches; the dataset has many spelling mistakes, with words interpreted differently and often wrongly by users not fluent in English. Often users only hear these words in the news, and are not even familiar with the spelling of the word in the original language. To construct a more realistic data set, we could ask Arabic writers to transliterate words from a recording; this would allow pronunciation to be accurately captured by users not fluent in English.

Information retrieval systems must cater for common spelling variants; our results help understand how to identify these in Arabic text.

References

Ahmed Abdelali, Jim Cowie, and Hamdy S. Soliman. 2004. Arabic information retrieval perspectives. In *Proceedings of the 11th Conference on Natural Language Processing, Journes d'Etude sur la Parole - Traitement Automatique des Langues Naturelles (JEP-TALN)*, Fez, Morocco.

Nasreen Abduljaleel and Leah S. Larkey. 2003. Statistical transliteration for English-Arabic cross-language information retrieval. In *Proceedings of the International Conference on Information and Knowledge Management*, pages 139–146, New Orleans, LA, USA. ACM Press.

Jamal B. S. Al-Qinal. 2002. Morphophonemics of loanwords in translation. *Journal of King Saud University*, 13:1–132.

Syed Uzair Aqeel, Steve Beitzel, Eric Jensen, David Grossman, and Ophir Frieder. 2006. On the development of name search techniques for Arabic. *Journal of the American Society for Information Science and Technology*, 57(6):728–739.

Christine L. Borgman and Susan L. Siegfried. 1992. Getty's synoname and its cousins: A survey of applications of personal name-matching algorithms. *Journal of the American Society for Information Science*, 43(7):459–476.

Lee R. Dice. 1945. Measures of the amount of ecologic association between species. *Ecology*, 26(3):297–302, July.

T. Gadd. 1990. Phonix: the algorithm. *Program*, 24(4):363–369.

Patrick A. V. Hall and Geoff R. Dowling. 1980. Approximate string matching. *ACM Computing Surveys*, 12(4):381–402.

Dan Melamed. 1995. Automatic evaluation and uniform filter cascades for inducing N-best translation lexicons. In David Yarovsky and Kenneth Church, editors, *Proceedings of the Third Workshop on Very Large Corpora*, pages 184–198, Somerset, New Jersey. Association for Computational Linguistics.

Abdusalam F Ahmad Nwesri, S. M. M. Tahaghoghi, and Falk Scholer. 2006. Capturing out-of-vocabulary words in Arabic text. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP 2006)*, pages 258–266, Sydney, Australia, 22–23 July. Association for Computational Linguistics.

Ulrich Pfeifer, Thomas Poersch, and Norbert Fuhr. 1996. Retrieval effectiveness of proper name search methods. *Inf. Process. Manage.*, 32(6):667–679.

Vijay Raghavan, Peter Bollmann, and Gwang S. Jung. 1989. A critical investigation of recall and precision as measures of retrieval system performance. *ACM Trans. Inf. Syst.*, 7(3):205–229.

Bonnie Glover Stalls and Kevin Knight. 1998. Translating names and technical terms in Arabic text. In *COLING/ACL Workshop on Computational Approaches to Semitic Languages*, pages 34–41, Montreal, Quebc, Canada.

Graham A Stephen. 1992. String search. Technical report, School of Electronic Engineering Science, University College of North Wales.

Esko Ukkonen. 1992. Approximate string-matching with q-grams and maximal matches. *Theor. Comput. Sci.*, 92(1):191–211.

Robert A. Wagner and Michael J. Fischer. 1974. The string-to-string correction problem. *J. ACM*, 21(1):168–173.

Justin Zobel and Philip Dart. 1995. Finding approximate matches in large lexicons. *Software - Practice and Experience*, 25(3):331–345.

Justin Zobel and Philip Dart. 1996. Phonetic string matching: lessons from information retrieval. In *The 19th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 166–172, New York, NY, USA. ACM Press.