Morphological Analysis of Ill-formed Arabic Verbs in Intelligent Language Tutoring Framework

Khaled Shaalan[1] Marwa Magdy[2] Aly Fahmy[2]

[1]The British University in Dubai, PO Box 345015 Dubai, UAE
[2]Faculty of Computers & Information, Cairo University, 5 Ahmed Zewel St., Giza 12613 Egypt
khaled.shaalan@buid.ac.ae, m.magdy@fci cu.edu.eg, a.fahmy@fci cu.edu.eg

Abstract

Arabic is a language of rich and complex morphology. The nature and peculiarity of Arabic make its morphological and phonological rules confusing for second language learners (SLLs). The conjugation of Arabic verbs is central to the formulation of an Arabic sentence because of its richness of form and meaning. In this paper, we address issues related to the morphological analysis of ill formed Arabic verbs in order to identify the source of errors and provide an in formative feedback to SLLs of Arabic. The edit distance and constraint relaxation techniques are used to demonstrate the capability of the proposed approach in generating all possible analyses of erroneous Arabic verbs written by SLLs. Filtering mechanisms are applied to exclude the irrelevant constructions and determine the target stem. A morphological analyzer has been developed and effectively evaluated using real test data. It achieved satisfactory results in terms of the recall rate.

1 Introduction

For a natural language, morphology is a basic layer over which higher syntactic and semantic layers are built (Jurafsky and Martin 2008; Darwish 2002). Most of applications concerned with the natural text-based human machine interaction depend on reliable morphological analysis systems. These systems cannot expect all their input to conform to the linguistic rules encoded in them. So, they should react, in some way, to ill-formedness (erroneous) input. The action taken by them varies from one application to another. A *spelling checker* highlights ill-formed input and tries to give possible alternative words. However, a *computer based language learning system* should be equipped with good diagnostics abilities to identify the source of error and provide an informative feedback to their users. Other applications should be able to either skip the error or provide some form of (normalized) output for this erroneous input (Faltin 2003).

Many research, however, have attacked the problem of Arabic morphological analysis (Ahmed 2000; Beesley 2001; Buckwalter 2002; Darwish 2002; Al-Sughaiyer and Al-Kharashi 2004; Attia 2006). But to the best of our knowledge few research have addressed the problem of analysis of ill-formed Arabic words (e.g., Bowden and Kiraz 1995; Ahmed 2000; Buckwalter 2002). Bowden and Kiraz (1995) investigated the problem of correcting words in Semitic languages including Arabic language. Their approach integrated with morphological analysis using a multi-tape formalism. The model had two-level error rules that handle the following error types: vowel shift, deleted consonant, deleted long vowels, and substituted consonant. Moreover, Ahmed (2000) and Buckwalter (2002) applied some spelling relaxation rules (to deal with orthographic variations like the use of the final letter ه /h/[1] instead of the letter ة /p/) to get all possible analyses of an erroneous word. However, these systems only handle performance errors made by native speakers of the language.

This paper addresses issues related to the morphological analysis of ill-formed Arabic verbs written by beginner to intermediate SLLs. The proposed system is an integral part of an intelligent language tutoring system for Arabic. The edit distance and constraint relaxation techniques are used to generate all possible analyses of erroneous Arabic verbs. Filtering mechanisms are applied after the extraction of affixes and stems to exclude the irrelevant constructions and determine the target stem. For each case, a morphological gloss is incrementally formulated which is to be used as a base for constructing the feedback to the learner.

The rest of this paper is structured as follows. Section 2 presents an overview of Arabic Morphology. Section 3 introduces an analysis of common Arabic lexical errors. Section 4 describes the proposed model. Section 5 discusses the results from an experiment. Finally, in Section 6 we give some concluding remarks.

[1] For transliteration, we refer the reader to Buckwalter (2002).

2 The Arabic Morphology System

Arabic language is one of the Semitic languages that is defined as a *diacritized* language where the pronunciation of its words cannot be fully determined by their spelling characters only. It depends also on some special marks put above or below the spelling characters to determine the correct pronunciation; these marks are called diacritics, so-called "Tashkil" in Arabic.

Unfortunately, in nowadays Arabic writing, people do not explicitly mention diacritics. They depend on their knowledge of the language and the context to understand none or partial diacitized text. Due to the optional diacritization, two or more words in Arabic are homographic: they have the same orthographic form, though the pronunciation and meaning is totally different Ahmed 2000; Attia 2006).

In addition to its orthographic nature, Arabic is rich in derivational morphology. It is considered a non-concatenative language as it alters the stem of words according to the syntactic context. Arabic words are morphologically divided into three types: noun, verb, and particle. They are derived by applying a pattern to the root to generate a *stem* and then inflect into prefixes and suffixes (Diab, Hacioglu, and Jurafsky 2004).

Consequently, the derivation process has made the Arabic language the richest in vocabulary ever found among all important natural languages (Ahmed 2000), although it has a relatively small number of derivative *patterns*. However, one of the most puzzling problems in the study of Arabic is its verbal system which is very rich in forms and meaning (Soudi, Cavalli, and Jamari 2001).

The conjugation of verbs in different tenses, voices and mood is achieved using well behaved morphological rules. The irregularities are due to the phonological constraints of certain root consonants. The important irregularity issues are related to Arabic weak verbs that include one or more weak letter. Weak letters can be deleted or substituted by other letters because of Arabic phonological constraints (El-Sadany and Hashish 1989). For example, the replacement of the letter (و) /w/ by (ا) in taking the past (perfect) tense of the trilateral root ق-و-ل /q-w-l/, using regular rules would generate *قولَ[2] /qawala/ but as it is a hollow (middle weak) verb it should be generated according to special weak rules and thus it appears in written texts as قالَ /q**A**la/ (said).

3 Arabic Lexical Error Typology

An important step in the implementation of an error analysis system is to decide which type of errors to be analyzed. Realistically, not every imaginable error type can be analyzed within a single system (Faltin 2003). There are two main criteria to select errors. On the one hand, errors which are easy to implement given the linguistic resources at hand and the diagnosis techniques available. On the other hand, there are the needs of the end-user population which makes specific kinds of errors.

To decide on the set of errors handled by our system, the Arabic SLLs needs were investigated by examining a set of linguistic studies which defined the most frequent types of errors made by SLLs (cf. Ali 1998; Abd Alghaniy 1998; Jassem 2000). Tables 1 through Table 2 provide details of possible errors which are commonly made by SLLs of Arabic. These errors are classified as *word formation errors* due to *morphology* and *phonology*.

Error Type	Source of Error
Connected pronouns Acronym: CP	Incorrect usage of pronouns with respect to verb tense **Example:** Wrong: **يجنت***[3] /ya-ji}o-tu/ (I-he-came) Correct: **جنت** /ji}o-tu/ (I-came)
Verb conjugation Acronym: VC	Incorrect conjugation of Arabic weak verbs **Example:** wrong: **نجو*** /najaw/ correct: **نجا** /najA/ (he escaped)

Table 1: Word Formation Errors due to Morphology

Error Type	Source of Error
Consonant letters Acronym: CL	Incorrect usage of letters with a closely related pronunciation **Example:** Wrong: **اصتطيع*** />a-SotaTiyE/ Correct: **أستطيع** />a-sotaTiyE / (I-am-able).
Vowel letters Acronym: VL	Making short vowel a long one **Example:** Wrong: **أصباحت*** />aSobAH-at/. Correct: **أصبحت** />aSobaH-at / (became)
	Making long vowel a short one **Example:** Wrong: **تزرين*** /ta-zuri-yna/. Correct: **تزورين** /ta-zwri-yna/ (you-visit)

Table 2: Word Formation Errors due to Phonology

Notice that according to Tschichold (2003) lexical errors are classified into *three* main classes:

- Errors in word formation. They are due to incorrect application of either morphological or phonological rules.
- Errors in word choice. They are due to the ambiguity in word senses and phonetics.
- Errors at the interface of lexical and grammar. They are due to the morpho-syntactic features of words.

The proposed system focuses on word formation errors. Other errors are out of this paper scope.

4 The Proposed Model

The proposed model depends on a set of linguistic studies which follow error analysis approach in identifying most frequent errors made by SLLs. This approach however follows some steps to identify and classify errors: data

[2] The asterisk indicates an incorrect word or sentence.

[3] These examples are collected from different real materials which are committed by different Arabic SLLs.

collection, error identification, error classification, error description, error explanation and pedagogical application (Jassem 2000). Therefore these studies collect real materials written by SLLs in a typical teaching/learning environment; these learners have different backgrounds (i.e., differ in their first language). Consequently, the extracted errors are general not aimed to a specific sort of learners. Therefore, the proposed model is general enough and can be used by any sort of learners.

However, the proposed model generates *all* possible word analyses for each ill-formed input. It uses *constraint relaxation* and *edit distance* techniques to split each erroneous word into three segments: *prefix+stem+suffix*. In any language model, the partial structures can combine only if some constraints or conditions are met. When these constraints are relaxed, an attachment is allowed even if the constraint is not satisfied (Faltin 2003).

In Arabic, various constraints should be met to formulate a well-formed word such as *usage of certain connected pronouns with respect to a verb tense* and *the usage of certain affixes or clitics with conjugated verbs*. In the proposed model, these two example constraints can be relaxed to allow for error diagnosis.

In general, the proposed model takes every erroneous input word and proceeds with the following steps to perform its functionality:

1. Extract a list of all possible suffixes.
2. Filter the suffixes list.
3. Extract a list of all possible prefixes.
4. Filter the prefixes list.
5. Construct all possible correct stems.
6. Form groups of similar stems.
7. Get the base word forms[4] from stem strings.
8. Match the correct answer base word form with the learner answer base word forms and determine the analyses of the ill-formed input.

These steps are necessary as the conjugated verb might be made ill-*formed* due to either ill-formed generation of a stem from applying a pattern to the root or ill-formed inflection of the stem with affixes. The following shows the application of these steps on Example 1.

Example 1:

Write a sentence using the following Arabic roots. ق و ل، ح ق، د و م */q w l, H q, d w m/.*

Assume the following two answers; where (a) includes a wrong conjugation of a *Hollow* (middle weak) verb, and (b) is the correct answer.

a. **قالتو*** الحق دائما /qAlo-tw AlHaq~ dA}imAF/ (I always told the-truth).

b. **أقول** الحق دائما />a-quwl AlHaq~ dA}imAF/ (I always tell the-truth).

The proposed model first matches the correct answer with the learner answer and filters out the matched words. This leaves the correct answer with the word أقول />a-quwl/ (I-tell) while it leaves the learner answer with the word قالتو /qAlo-tw/ (told-I). Then the model applies all the previous steps on the word قالتو*.

Step 1: Extract a list of all possible suffixes

The model uses regular expressions for representing the list of affixes to be extracted. The regular expressions are implemented using the deterministic finite-state automata (FSA) approach. For more information about FSA, see (Jurafsky and Martin 2008). The suffix list is represented in the deterministic FSA in reverse order to facilitate left to right matches. Figure 1 illustrates a FSA representation of the suffixes: وا /Waw-Alef, Alef-Teh, Alef, Teh /.ت, ا, ات

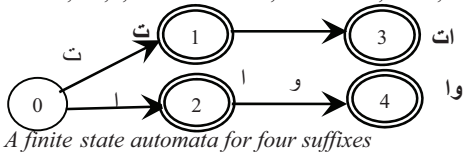
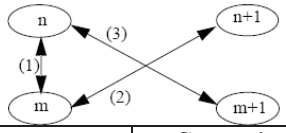


Figure 1: A finite state automata for four suffixes

To extract the suffix list, the system matches the input word against the suffix automata. The match begins at the *end* of the word (Position 1) and works *backwards*. The system relaxes the *usage of certain affixes* constraint by using a *three way match* technique (Elmi and Evens 1998) to compare two strings: a suffix of the learner input with a legal suffix. This method assumes that when a character at location *n* of the first string does not match a character at location *m* of second string, there exist an error and two other comparisons are made (character at position *n* with character at position *m+1* and character at position *n+1* with character at position *m*); Initially, *n=1* to point to the last letter of the input string and *m=0* to point to a letter at the initial state in the FSA. The three-way-match comparison and the order of the comparisons are shown in Figure 2.



Results of Comparison	Comparison number		
	1	2	3
No error	**T**		
Extra character	F	**T**	F
Missing character	F	F	**T**
Character substitution	F	F	F

Figure 2: The three way match comparison and the order of the comparison

Given the FSA at Figure 1 and a learner response with the word قالتو* /qAlo-tw/ (told-I), the system tries to: 1) match the last letter (*n=1*) و /w/ of the input word with the Arabic suffix ت (Teh) that occurs at the end (*m=1*) of

[4] The *base word* form is a, normalized, stem form after removing all weak and hamza letters to facilitate the matching of different verb conjugations of the same root without taking into consideration the lexicographic change that may happen to these irregular forms.

Arabic verbs. The match fails. So, it tries to match again with the one but last letter ($n+1=2$) of the input word ت /t/ which succeeds. This process interprets the letter ت /t/ as a possible suffix and the letter و /w/ as an extra letter occurring at the end of the input word. Similarly, the match exhaustively proceeds with other Arabic suffixes yielding the following **10** possible solutions along with their error indications, respectively:

1. [""], NULL suffix.
2. ["ات"], Feminine plural noun suffix with extra Waw and missing Alef.
3. ["ت"], Third person singular feminine perfect verb suffix with extra Waw.
4. ["ت"], First person singular perfect verb suffix with extra Waw.
5. ["ت"], Second person singular feminine perfect verb suffix with extra Waw.
6. ["ت"], Second person singular masculine perfect verb suffix with extra Waw.
7. ["و"], Masculine plural noun suffix.
8. ["وا"], Second person masculine plural imperative verb suffix with missing Alef.
9. ["وا"], Masculine plural imperfect verb suffix with missing Alef.
10. ["وا"], Third person masculine plural perfect verb suffix with missing Alef.

Practically, however, the use of constraint relaxation in analyzing Arabic verbs leads to over-generation. In order to resolve this issue, we introduced *heuristic rules* that eliminate highly implausible analyses made by Arabic SLLs. For example, SLLs of Arabic might find it difficult to choose among a vowel sign such as (الضمة) /u/ and a genuine character, such as letter و /w/. So, one set of the heuristic rules restricts itself to handle the extra or missing weak letters. Another set of rules restricts itself to recognize a letter substituted by another similar letter in pronunciation. The closely related pronunciation letters are categorized into 9 groups:1) ['ض', 'د', 'ت', 'ط'], 2), ['ص', 'س', 'ز'], 3) ['س', 'ث'], 4) ['ق', 'ج'], 5) ['ق', 'ك'], 6) ['ظ', 'ز', 'ذ'], and 7) ['ح', 'ع'], 8).

Step 2: Filter the suffixes list

This step excludes some irrelevant suffixes according to: learner's answer and the set of error categories handled by the proposed system. For example, the previous list of 10 suffixes could be minimized to **five** solutions: 1, 4, 8, 9, and 10. There are two explanations behind this filtering. The system does not handle errors related to Arabic nouns which led to ignore the *second* and *seventh* solutions. Second, the other three eliminated solutions are discarded since their end case does not match the extra character و /w/.

Step 3: Extract a list of all possible prefixes

Extracting the prefixes list is the same as extracting the suffixes list except that the order of the match process begins at the *first* letter and proceeds *upwards*. Applying this step on Example 1 produces only null prefix solution:

1. [""], NULL prefix.

Step 4: Filter the prefixes list

As the prefixes list is null, the output of this step does not result in any filtered prefix.

Step 5: Construct all possible correct stem forms

To construct a possible correct stem, the system tries exhaustively to extract every possible stem (i.e., a substring that remains after removing prefixes and suffixes from the input word) such that either the compatibility conditions between affixes[5] is satisfied or the relaxed constraints are met. In Arabic, there are certain connected pronouns that can only be used with a certain verb tense such as the suffix pronoun 'نا' (Na) which can only be used with the perfect tense while the prefix pronoun 'نـ' (Noon) which can only be used with the imperfect tense. It is morphologically incorrect to use both pronouns at the same time, e.g. 'نذهبنا'*, as this will be considered as a severe contradiction in pronoun inflections; hence, leads to a conflict in verb tense. Applying constraint relaxation technique will split this erroneous word into: the prefix 'ن', the stem 'ذهب', and the suffix 'نا' even though the attachment constraint is not met.

The output of this step, using the results so far from Example 1, yields **four** solutions. Each solution consists of five elements constituting: *prefix*, *stem*, *suffix*, *feature structure (FS)*[6] that describes the analyzed word, and an initial *error indication*. The *error indication* is a list that denotes: the required operation (e.g., insert) to relax the affix, the actual character and the position where the operation should take place.

Solution (1):

- *Null affixes*[7]: Null Prefix + "قالتو" + Null suffix
  Error indication: [];

Solution (2):

- *Null prefix with first person singular perfect verb in active voice with extra Waw in the suffix*:
  Null Prefix + "قال" + 'ت'
  Error indication: [insert('و',5)];

Solution (3):

- *Null prefix with second person masculine plural imperative verb with deleted Alef in the suffix*:
  Null Prefix + "قالت" + 'وا'
  Error indication: [delete('ا',6)];

Solution (4):

- *Null prefix with third person masculine plural perfect verb in active voice with deleted Alef in the suffix*:
  Null Prefix + "قالت" + 'وا'
  Error indication: [delete('ا',6)]

Notice that, in this example, the solution *"Null prefix with masculine plural imperfect verb with deleted Alef in the suffix"* is discarded as the combination between a null

[5] The compatibility table is taken from (Buckwalter 2002).

[6] The FS includes the following features: *lexical category*, *pattern*, *tense*, *voice*, *mood*, *subject* and *object person*, *number*, *gender*, which is not shown due to space limitation.

[7] This solution represents a perfect verb in the third person singular masculine active voice.

prefix with the imperfect suffix 'وا' (Waw & Alef) (cf. the 9th suffix solution in *step 1*) is morphologically invalid (incompatible). This suffix can only be used together with either one of the following prefixes: 'ي' (Yeh) or 'ت' (Teh).

Step 6: Form Groups of similar stems

The current solution list may contain similar stem strings. So, in order to avoid redundancy, the list is organized into groups of lexicographically similar stems. The output of this step yields three groups in the solution stem list: {('قالتو', [1]), ('قال', [2]), ('قالت', [3, 4])}; where the number points to the corresponding five elements in the solution list.

Step 7: Get the base word forms from stem strings

To get all possible base forms (normalized stems) from each string in the stem solution list, we need first to match with a list of Arabic verb patterns. These patterns are represented as deterministic FSA. Figure 3 illustrates a FSA of the relevant patterns فعل, فاعل, تفعل, تفاعل /tafAEala, tafoE~al, faEala, fAEala/. We differentiate between two types of characters in a pattern: *fixed* and *generic*. A generic character can represent any Arabic letter while a fixed character should represent an exact same character. For example, the pattern فعل /faEala/ has only three generic characters while the pattern تفعل /tafoE~al/ has one fixed character ت /t/ and three generic characters.

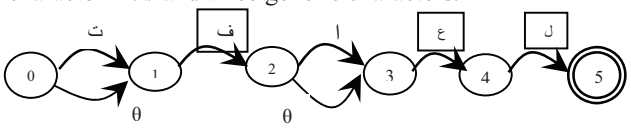


Figure 3: A finite state automata for four Arabic verb patterns. The letter inside a square is generic.

The system matches characters of the stem string against characters of the verb pattern using the *three way match* technique but to relax only the missing and substituted letters that are similar in pronunciations. If a match succeeds, the resultant word is normalized by deleting any weak or hamza letters and; then the obtained base word form is included in the base form solution list.

This step is applied to the current stem solution list {('قالتو', [1]), ('قال', [2]), ('قالت', [3, 4])}. The first stem in this list is discarded as it does not match with any Arabic pattern. The processing of the second stem produces the base forms: {'قال' and 'قل'}; which after removing the middle weak letter, i.e. *Alef*, of the first one it becomes normalized to the second (i.e., 'قل'). The processing of the third solution produces the base forms {'قالت' and 'قلت' which is similarly normalized to 'قلت'. Ultimately, the base form solution list consists of the base forms {('قل', [2]), ('قلت', [3, 4])} (cf. the 2nd, 3rd and 4th solutions in *step 5*).

Step 8: Match the correct answer base word form with the learner answer base word forms and determine the analyses of the ill-formed learner input.

This step obtains first the base word forms of the roots stored with the question. Then, it matches each of these base forms with each base form in the learner's answer. This process is deterministic such that once a match is found all other forms from the solution list are discarded and the final word analysis is generated.

Applying this step on the base form solution list, the match succeeds with the first base word form[8] (i.e. 'قل'). This yields the final solution "*Null prefix with first person singular perfect verb in active voice with extra Waw in the suffix*" as the only possible word analysis for the erroneous word قالتو* / qAlo-tw / (told-I).

A comparison between the features of the correct word أقول />a-quwl/ (I-tell) and the features derived from the analysis of the incorrect word قالتو* /qAlo-tw/ (told-I) shows that the learner has made three errors: 1) *Verb tense* error since the correct tense is *imperfect* while the incorrect one is *perfect*, 2) *short vowel substituted by long vowel* error since there is an extra *Waw* character in affix representation, and 3) *Verb conjugation* error since there is an extra character at position 2 of the stem قال /qAla/ and this character does not match the diacritic sign of the correct word which is ضمة /u/. The system will provide an appropriate feedback describing these errors.

5 Experiment

We conducted an experiment that measures how successfully the proposed model generates *all* possible analyses of erroneous Arabic verbs that are used later to diagnose SLLs errors. The *quantitative* measures are used. These measures rely on collecting different test sets written by real SLLs in a typical teaching/learning environment. It was necessary that these learners have different backgrounds (i.e., differ in their first language) to test if the system is general enough and not aimed to a specific sort of learners. The different types of errors and the exact source of errors in the test set are *subjectively* identified by a human specialist to produce the reference set. The test set is then fed into the morphological analyzer and the detected and undetected errors are reported. These errors are based on analyses generated by the proposed model. The recall rate[9] for each error type is calculated. This measure has been used in evaluating similar research (cf. Wagner, Foster, and Genabith 2007; Sjöbergh and Knutsson 2005; Faltin 2003).

The above mentioned methodology is applied on a real test set that consists of 116 real Arabic sentences. The number of words per sentence varies from 3 to 15 words, with an average of 5.1 words per test sentence. The total number of words in all test sentences are 587 words, 118 of them have lexical verb errors. However, 60 of erroneous verbs are *word formation* errors. Others are either word

[8] The base word form of the correct root ق و ل /q w l/ is قل after removing the weak letter و /w/

[9] The percentage of each error type in the test set that actually diagnosed by the system.

choice or error at the interface between lexical and grammar which are irrelevant to this paper. Table 3 summarizes the evaluation results.

Error Type	N	fully Diagnosed		General Error indication	
		N	%	N	%
CL	8	8	100	0	0
VL	24	19	79.2	5	20.8
VC	21	14	66.7	7	33.3
CP	7	6	85.7	1	14.3
Total	60	47	78.3	12	20

Table 3: evaluation results

The last column in Table 3 shows the cases of general error indication (i.e. the system fails to detect the exact source of error the learner made). These cases arose because of ambiguity; the system does not have enough knowledge of what the student meant to express. For instance, consider the erroneous word أجوب. It is not clear whether the learner meant the word to be: 1) the imperfect verb أجيب />u-jiyb/ (I-answer), 2) or imperfect verb أجوب />a-juwb/ (I-explore).

6 Conclusion

Arabic is a highly derivational language that makes it a challenge to SLLs. Therefore, SLLs not only make errors done by native speakers but also others that arise due to competence issues. Consequently, using methods and tools designed for a native speaker spell checking is not a good way to proceed, especially for highly derivational and inflectional languages such as Arabic. Therefore, the nowadays methods and tools should be refined to meet the SLLs needs. In the absence of a complete computationally erroneous Arabic corpus that can be used to evaluate the proposed model, we have to manually collect the test set from the real teaching environment. The test set was relatively small but it was sufficient to show that the approach and techniques employed in this paper have successfully generated all possible analyses of ill-formed verbs written by SLLs of Arabic, in particular, when it comes to difficult constructions such as Arabic weak verbs. From a pedagogical point of view, the achieved rich analyses enable feedback elaboration that helps learners to understand better their knowledge gab.

References

Abd Alghaniy, K. E. 1998. Arabic and Malaysian Languages from Phonological and Morphological Perspective: A Contrastive Analysis Approach. Master thesis, Cairo University, Egypt.

Ahmed, M. A. 2000. A Large Scale Computational Processor of the Arabic Morphology, and Applications. Master thesis, Cairo University, Egypt.

Ali, M. B. 1998. Linguistic Analysis of Mistakes by Students at the University of Malaya: An Error Analysis Approach. Master thesis, Cairo University, Egypt.

Al Sughaiyer, I. A., and Al Kharashi, I. A. 2004. Arabic Morphological Analysis Techniques: A Comprehensive Survey. In American Society for Information Science and Technology Journal, 55(3): 189 213.

Attia, M. A. 2006. An Ambiguity Controlled Morphological Analyzer for Modern Standard Arabic Modeling Finite State Networks. In Proceedings of the Challenge of Arabic for NLP/MT Conference, 2006. The British Computer Society, London.

Beesley, K. R. 2001. Finite State Morphological Analysis and Generation of Arabic at Xerox Research: Status and Plans in 2001. In Proceedings of the Arabic Language Processing: Status and Prospect, (ACL 2001). Toulouse, France, pp: 1 8.

Bowden, T., and Kiraz, G. A. 1995. A Morphographemic Model for Error Correction in Non concatenative Strings. In Proceedings of ACL 1995, Boston, Massachusetts, pp: 24 30.

Buckwalter, T. 2002. Buckwalter Arabic Morphological Analyzer Version 1.0. Linguistic Data Consortium, University of Pennsylvania, LDC Catalog No.: LDC2002L49, ISBN 1 58563 257 0.

Darwish, K. 2002. Building a Shallow Morphological Analyzer in One Day. In Proceedings of the Workshop on Computational Approaches to Semitic Languages, (ACL 2002), Philadelphia, PA, USA, pp: 47 54.

Diab, M., Hacioglu, K., and Jurafsky, D. 2004. Automatic Tagging of Arabic Text: From Raw Text to Base Phrase Chunks. In Proceedings of HLT NAACL 2004, Boston, MA, pp: 149 152.

Elmi, M. A., and Evens, M. 1998. Spelling Correction Using Context. In Proceedings of ACL 1998, Montreal, Canada, pp: 360 364.

El Sadany, T. A. and Hashish, M. A. 1989. An Arabic Morphological System. In IBM Systems Journal, 28(4): 600 612.

Faltin, A. V. 2003. Syntactic Error Diagnosis in the Context of Computer Assisted Language Learning. PhD thesis, University of Geneva, Switzerland.

Jassem, J. A. 2000. Study on Second Language Learners of Arabic: An Error Analysis Approach. Kuala Lumpur (Malaysia): A.S. Noordeen. ISBN 983 065 093 6.

Jurafsky, D., and Martin, J. H. 2008. Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistic and Speech Processing. Prentice Hall Series in Artificial Intelligence. ISBN 978 013 187 321 6

Sjöbergh, J., and Knutsson, O. 2005. Faking Errors to Avoid Making Errors: Machine Learning for Error Detection in Writing. In Proceedings of RANLP 2005, Borovets, Bulgaria, pp: 506 512.

Soudi, A., Cavalli Sforza, V., and Jamari, A. 2001. A Computational Lexeme based Treatment of Arabic Morphology. In Proceedings of the Workshop on Arabic Language Processing: Status and Prospects, (ACL 2001), Toulouse, France, pp: 155 162.

Tschichold, C. 2003. Lexically Driven Error Detection and Correction. In CALICO Journal, 20(3): 549 559.

Wagner, J., Foster, J., and Genabith, J. V. 2007. A Comparative Evaluation of Deep and Shallow Approaches to the Automatic Detection of Common Grammatical Errors. In Proceedings of EMNLP CoNLL 2007, Prague, Czeck Republic, pp: 112 121.