Analysis and feedback of erroneous Arabic verbs

KHALED SHAALAN¹, MARWA MAGDY² and ALY FAHMY³

¹School of Informatics, University of Edinburgh, Edinburgh, UK; The British University in Dubai, P.O. Box 345015, Dubai, UAE
²IBM Egypt – Cairo Technology Development Centre, Pyramids Heights Office Park, Building C10, Cairo–Alexandria Desert Road, KM 22, P.O. Box 166, El-Ahram, Giza, Egypt
³Faculty of Computers & Information, Cairo University, 5 Ahmed Zewel St., Giza 12613, Egypt

(Received 2 December 2012; revised 28 June 2013; accepted 26 July 2013; first published online 16 September 2013)

Abstract

Arabic language is strongly structured and considered as one of the most highly inflected and derivational languages. Learning Arabic morphology is a basic step for language learners to develop language skills such as listening, speaking, reading, and writing. Arabic morphology is non-concatenative and provides the ability to attach a large number of affixes to each root or stem that makes combinatorial increment of possible inflected words. As such, Arabic lexical (morphological and phonological) rules may be confusing for second language learners. Our study indicates that research and development endeavors on spelling, and checking of grammatical errors does not provide adequate interpretations to second language learners’ errors. In this paper we address issues related to error diagnosis and feedback for second language learners of Arabic verbs and how they impact the development of a web-based intelligent language tutoring system. The major aim is to develop an Arabic intelligent language tutoring system that solves these issues and helps second language learners to improve their linguistic knowledge. Learners are encouraged to produce input freely in various situations and contexts, and are guided to recognize by themselves the erroneous functions of their misused expressions. Moreover, we proposed a framework that allows for the individualization of the learning process and provides the intelligent feedback that conforms to the learner's expertise for each class of error. Error diagnosis is not possible with current Arabic morphological analyzers. So constraint relaxation and edit distance techniques are successfully employed to provide error-specific diagnosis and adaptive feedback to learners. We demonstrated the capabilities of these techniques in diagnosing errors related to Arabic weak verbs formed using complex morphological rules. As a proof of concept, we have implemented the components that diagnose learner’s errors and generate feedback which have been effectively evaluated against test data acquired from real teaching environment. The experimental results were satisfactory, and the performance achieved was 74.34 percent in terms of recall rate.

1 Introduction

There are many reasons that attract people to learn Arabic. These reasons include its importance for culture, religion, and economics. Some people also want to learn...
it just for the sake of learning a second language. Arabic is very rich in morphology and syntax. As such, learners of Arabic require some form of feedback, which can be described as a reaction to what has been said or written. This feedback most often comes from other human beings with whom the language learner is interacting. There are, however, other means to receive feedback. The recent trend is to automate the process using Intelligent Language Tutoring System (ILTS) software (Shaalan 2005a). This software contains various Arabic computational tools that help learners study Arabic lessons, improve their Arabic language skills, understand Arabic morphology and grammar, and practice exercises, among others. In order to provide Arabic learners with proper feedback, their responses or answers should be analyzed by tools capable of diagnosing errors. So, it is necessary to include a natural language processing (NLP) component in Arabic ILTS.

There are two common Arabic NLP approaches, namely rule-based and corpus-based (or statistical-based) approaches (Abdel Monem et al. 2008; Shaalan 2010). The rule-based approach relies on handcrafted morphological or grammatical rules written by linguists. The main advantage of the rule-based approach is that it is based on a core of solid linguistic knowledge. Moreover, it is easy to incorporate techniques for diagnosing learner errors that provide highly accurate feedback for these errors. However, any maintenance or update applied to the rule-based component is labor-intensive and time-consuming, especially if the linguists with the required knowledge and background are not available. On the other hand, the corpus-based approach utilizes learning algorithms that require large tagged learner corpus for training and testing. Learning algorithms involve a selected set of features extracted from datasets annotated with linguistics knowledge in order to generate statistical models that are used for learner’s error prediction. An advantage of the corpus-based approach is that it is adaptable and updatable with minimal time and effort as long as sufficiently large datasets are available. It is worth noting that the lack of linguistic resources creates a critical obstacle when it comes to Arabic NLP in general and Arabic error diagnosis in particular.

The current work, we call it Arabic ILTS, concentrates on NLP tools and techniques geared toward the diagnosis of Arabic lexical errors produced by Second Language Learners (SLLs) in the context of the ILTS framework. The rule-based approach is used in our research to diagnose these errors. The main reasons for our choice of the rule-based approach are the lack and limitations of Arabic linguistic resources (i.e. sufficiently large erroneous learner corpus), the success in developing efficient techniques for handling error analysis and corrections (e.g. constraint relaxation and edit distance techniques), and the high performance of NLP systems that have employed rule-based transformations (Shaalan 2010).

Arabic has several morphological properties that make error diagnosis and feedback challenging. In this paper we consider the value of morpho-lexical and morpho-syntactic features such as stem and lexical category, respectively, which allow us to adopt error analysis and correction techniques in error diagnosis and feedback. Arabic words can be expressed in terms of their morphemes. Various lexical errors can be made by learners of Arabic. In addition to errors in the concatenative prefixes and suffixes, errors might be made in the generation of the templatic morpheme
Analysis and feedback of erroneous Arabic verbs

(stem) compose of a root and pattern. We successfully used constraint relaxation and edit distance techniques for performing Arabic word analysis and diagnosis (cf. Magdy, Shaalan and Fahmy 2007; Shaalan, Magdy and Fahmy 2010a, 2012; Shaalan, Magdy and Samy 2010b; Shaalan and Magdy 2011). These techniques are best clarified by examples. Consider the Arabic morphological rules that require certain connected pronouns to be used with specific verb tense such as the suffix pronoun ‘ة’ (Na – 1st pers, pl) which can only be used with perfect1 form, whereas the prefix pronoun ‘ن’ (Noon – 1st pers, pl) can only be used with the imperfect form. So if we want to express ‘we go’ ‘ذهبت’ /na-habu/ (we-go), we use the prefix ‘ن’ /n/ with the verb ‘go’ ‘ذهبت’ but to express ‘we went’ ‘ذهبت’ /ahab-nA/ (went-we), we use the suffix /nA/. It is morphologically incorrect to say ‘ذهبتنا’ (we-go-went-we) with both affixes appearing at the same time, as this is a severe contradiction in pronoun attachment that leads to conflict in verb tense. Applying constraint relaxation and edit distance techniques will allow us to split the erroneous word into the prefix ‘ن’ + the stem ‘ذهبت’ + the suffix even though the attachment constraint is not met (cf. Madgy et al. 2007; Shaalan et al. 2010a, 2010b, 2012; Shaalan and Magdy 2011). According to Arabic morphology, partial structures can combine only if some constraints or conditions are met. However, the constraint relaxation technique allows the constraints to be relaxed such that attachment is allowed even if a constraint is not satisfied. This technique is very useful to be able to split the erroneous word into three segments: prefix + stem + suffix. The edit distance technique is based on the number of edits (i.e. insertions, deletions, and substitutions) it takes to convert a wrong learner’s answer into a correct form. For example, the correct form ‘ذهبتِ’ (I went – fem) /ahab-tu/ of the erroneous word /‘ذهبتتو’ /ahab-tw/ can be identified by removing the extra ‘ة’ (Waw).

Language learners,3 however, can violate linguistic expectations at all levels: lexical, syntactic, semantic, and pragmatic/discourse. Traditional error analysis studies do not address lexical errors well enough, although there are a large number of word-related errors that are committed by non-native language users (Tschichold 2003). Many researchers, however, have attacked lexical error analysis in a variety of languages for the past few years, but to the best of our knowledge there is no deeply focused research on SLLs of Arabic. Some difficulties have contributed to the slow development of this research area: (1) Availability of linguistic resources, such as erroneous learner corpora acquired from real learning environment as well as their impact on favoring rule-based approach over corpus-based approach, (2)

1 Traditionally, Arabic grammarians classify Arabic verbs into three categories (Perfect, Imperfect, and Imperative; see Ryding 2005). The perfect tense refers to a completed action, i.e. a verb in the past tense. The imperfect category combines both present and futures tenses. Future is formed by adding the prefix ‘س’ to the present form of the verb. The other category, Imperative ‘ الأمر’ is not more than a mood for issuing orders or commanding rather than a tense, which is not different from negation or exclamation.

2 For transliteration, we refer the reader to Buckwalter (2002). Please note that this method includes the use of symbols such as " and as well as letters.

3 Faltin (2003) defined language learner as the person who does not speak a language with the fluency of a native speaker and actively seeks to improve his/her competence of the language. Competence means knowledge of the language – its rules of grammar and its vocabulary – all pieces of language and how they fit together (Jassem 2000).
availability of tools that support Arabic lexical error analysis and feedback and their impact on the speed of developing systems that improve the SLL knowledge, and (3) progressing that is sometimes hampered by linguistic (philosophical) issues that pop up frequently. This peculiarity of the Arabic language poses a great challenge for us to develop our lexical error diagnosis approach that addresses the word formation problem usually faced by SLLs of Arabic (Madgy et al. 2007; Shaalan and Magdy 2011; Shaalan et al. 2010a, 2010b, 2012). This is achieved by proposing a novel model that analyzes the learner’s answer and provides him/her with some form of feedback that identifies the source of the lexical error – related to Arabic verbs – s/he might made. Verbs play a central role within the Arabic sentence to which other sentence constituents are connected (Abdel Monem et al. 2008). Moreover, it is also the case that treatment of the verb phrase in Arabic grammar books takes up a lot of attention. In the linguistics literature, studies have indicated that errors in verbs tend to be rated as serious (Jassem 2000; Ali 2013).

An additional advantage that the current research provides is the individualization of the learning process. For example, when an SLL of Arabic responds to a question by writing an incorrect verb, the system is capable to distinguish between the following lexical error types: lexical category selection, pattern selection, tense selection, mood selection, subject–verb agreement, verb conjugation, connected pronouns and/or consonant, and vowel letters. Moreover, it is also capable to provide the adaptive intelligent feedback that conforms to the learner’s expertise for each class of error. Inexperienced learners might require detailed instruction, while experienced learners benefit from higher level reminders and explanations (Heift 1998b; Heift et al. 2000).

The rest of this paper is structured as follows. Section 2 presents a brief background on Arabic verbal system. Section 3 presents a review of lexical errors analysis problem. Section 4 introduces our study on the analysis of common Arabic lexical errors. Section 5 presents some examples of adaptive feedback that supports improving the learner’s linguistic knowledge. Section 6 describes an overall architecture of our proposed Arabic ILTS. Section 7 focuses on details of error diagnosis techniques developed and implemented within our system. Section 8 discusses how we evaluated our system. Finally, Section 9 presents concluding remarks and ideas for future research.

2 The Arabic verbal 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 (Ryding 2005). It also depends on some special marks put above or below the spelling characters to determine the correct pronunciation; these marks are called diacritics.

Arabic verb morphology is central to the generation of an Arabic sentence because of its richness of form and meaning (Habash 2010). Arabic verbs, however, can be conjugated from either trilateral or quadrilateral roots according to one of the traditionally recognized patterns (or conjugations). There are fifteen trilateral forms
Table 1. Arabic verbal stems

(a) Trilateral verbs

<table>
<thead>
<tr>
<th>Form Number</th>
<th>Pattern</th>
<th>Active Voice Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>“CVCVC”</td>
<td>/$ariba/ (drank)</td>
</tr>
<tr>
<td>2</td>
<td>“CVCCVC”</td>
<td>/kas~ara/ (shattered)</td>
</tr>
<tr>
<td>3</td>
<td>“CVVCVC”</td>
<td>/DAEafa/ (doubled)</td>
</tr>
<tr>
<td>4</td>
<td>“VCCVC”</td>
<td>/arAHa/ (brought relief)</td>
</tr>
<tr>
<td>5</td>
<td>“tVCCVC”</td>
<td>/taEal~ama/ (studied)</td>
</tr>
<tr>
<td>6</td>
<td>“tVCVVCVC”</td>
<td>/tasAqaTa/ (collapsed - fall piece by piece)</td>
</tr>
<tr>
<td>7</td>
<td>“nCVCVC”</td>
<td>/inokasara/ (broke)</td>
</tr>
<tr>
<td>8</td>
<td>“tVCVC”</td>
<td>/iqotafaY/ (follow)</td>
</tr>
<tr>
<td>9</td>
<td>“stVCCVC”</td>
<td>/isotagAva/ (asked for help)</td>
</tr>
<tr>
<td>10</td>
<td>“CCVCVC”</td>
<td>/iHomar~a/ (turned red)</td>
</tr>
<tr>
<td>11</td>
<td>“CCVVCVC”</td>
<td>/izoraAq~/ (became blue)</td>
</tr>
<tr>
<td>12</td>
<td>“CCVwCVC”</td>
<td>/igoraworaqa/ (immersed)</td>
</tr>
<tr>
<td>13</td>
<td>“CCVwVC”</td>
<td>/ijolaw~za/ (lasted long)</td>
</tr>
<tr>
<td>14</td>
<td>“CCVnCVC”</td>
<td>/iHolanokaka/ (turned jet black)</td>
</tr>
<tr>
<td>15</td>
<td>“CCVnCVy”</td>
<td>/iHobanoTay/ (caused swollen)</td>
</tr>
</tbody>
</table>

(b) Quadrilateral verbs

<table>
<thead>
<tr>
<th>Form Number</th>
<th>Pattern</th>
<th>Active Voice Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>“CVCCVC”</td>
<td>/daHoraja/ (rolled)</td>
</tr>
<tr>
<td>2</td>
<td>“tVCVCCVC”</td>
<td>/tazalozala/ (quaked)</td>
</tr>
<tr>
<td>3</td>
<td>“CCVnCVC”</td>
<td>/iforanoqaEa/ (dismissed)</td>
</tr>
<tr>
<td>4</td>
<td>“CCVCCVC”</td>
<td>/iDomaHal~a/ (faded away)</td>
</tr>
</tbody>
</table>

(The first nine forms are very common, while the rest are very rare) and four quadrilateral ones. Examples of all Arabic forms are shown in Table 1 (cf. Bowden and Kiraz 1995).

Arabic verbs appear in three tenses (perfect, imperfect, and imperative), two voices (active and passive), and four moods (indicative, subjunctive, jussive, and energetic). 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 (El-Sadany and Hashish 1989; Soudi, Cavalli-Sforza and Jamari 2001; Farghaly and Shaalan 2009). Arabic weak letters can be deleted or replaced by other letters because of Arabic phonological constraints. Arabic verbs can mainly be categorized into weak and strong verbs. These will be presented in the following sections.

2.1 Strong verbs

Strong verbs are those without any weak radical. They can be categorized into three classes: regular, hamzated, and doubled. Regular and doubled verbs can be generated using well-behaved morphological regular rules, while hamzated verbs are irregular ones. The hamza letter (حاء) is changed to other different realizations due to the influence of the vowels before and after the hamza (Al-Jumaily et al. 2011). The
different realizations of the hamza letter are (",ء،ئ،٪،ً) />, <, &, ,'}/. For example, the conversion of (i) />/ into (3) /&/ can be explained by taking the imperfect tense with passive voice for the root ﺲ-ل-ل />-k-l/. Applying regular rules would generate ﺲ-و-و /yu->okal/ but as a hamzated verb it should be generated as ﺲ-و-و /yu-&okal/ (be eaten).

### 2.2 Weak verbs

Weak verbs, a major concern of this work, can be categorized into three classes: **assimilated, hollow, and defective.**

**Assimilated verbs** are the verbs with a weak initial radical. They fall into two categories: verbs beginning with letter (3) /y/ such as "بَيْسَ" /ya'isa/ (despaired), and verbs beginning with letter (3)/ w /such as "وَعَدَ" /waEada/ (promised). The perfect and imperfect stems of the first category are generated by the same rules of strong verbs, while the imperfect stem of the second category is generated by the following specific rule:

- The initial radical (3) /w/ is deleted in the imperfect form if its middle short vowel is (i). For example, the imperfect conjugation of the root وَرْث /w-r-v/ is نَوْث /ya-riv/ (he/it inherits), while for the root دِجْل /w-j-l/ it is يَجْل /ya-wjal/ (he/it is afraid).

**Hollow verbs** are the verbs with a weak middle radical. They fall into the following four categories (Cavalli-Sforza, Soudi and Mitamura 2000; Soudi, Cavalli-Sforza and Jamari 2001):

- Verbs where the middle radical is (3) /w/ of the pattern ﺖ-ا-ا /faEala/ such as /Sawama/ ‘fasted’ or ﺖ-ا-ا /faEula/ such as ﺖ-ا-ا /Tawula/ ‘he/it was long’.
- Verbs where the middle radical is (3) /w/ of the pattern ﺖ-ا-ا /faEila/ such as ﺖ-ا-ا /xawifa/ ‘was afraid or frightened’.
- Verbs where the middle radical is (3) /y/ of the pattern ﺖ-ا-ا /faEala/ such as ﺖ-ا-ا /sayara/ ‘walked’.
- Verbs where the middle radical is (3) /y/ of the pattern ﺖ-ا-ا /faEila/ such as ﺖ-ا-ا /Hayira/ ‘was confused or hesitated’.

In both perfect and imperfect tenses, the stem is realized by either a short or a long vowel depending on person, number, and gender features (Cavalli et al. 2000). For example, to generate the third person masculine plural imperfect stem of the root صَوْم /S-w-m/, a long vowel is used such as ﺖ-ا-ا /ya-Suwmu-wna/ (they fast), while to generate the third person feminine plural imperfect stem of the same root, a short vowel is used ﺖ-ا-ا /ya-Sumo-na/ (they-(f) fast).

**Defective verbs** are the verbs with a weak final radical. They fall into the following five categories:

- Verbs where the final radical is (3) /w/ of the pattern ﺖ-ا-ا /faEala/ such as ﺖ-ا-ا /najawa/ ‘was rescued’.
- Verbs where the final radical is (3) /w/ of the pattern ﺖ-ا-ا /faEila/ such as ﺖ-ا-ا /raDiwa/ ‘was pleased’.
Analysis and feedback of erroneous Arabic verbs

• Verbs where the final radical is (۰) /w/ of the pattern /faEula/ such as /saruwa/ ‘was noble’.
• Verbs where the final radical is (۰) /y/ of the pattern /faEala/ such as /ramaya/ ‘threw’.
• Verbs where the final radical is (۰) /y/ of the pattern /faEila/ such as /xaziya/ ‘was despicable or ashamed’.

In both perfect and imperfect tenses, the stem is realized by either a short or a long vowel depending on the person, number, and gender features. For example, to form the third person masculine plural perfect stem of the root /n-j-w/, a short vowel is used such as /naja-woA/ (they were rescued), while to form the first person singular perfect stem of the same root, a long vowel is used /najawo-tu/ (I was rescued). The imperative tense can be generated from imperfect tense by adding /laf/ at the beginning of a verb.

3 Investigation of lexical error analysis and correction

Lexical (word) analysis is a fundamental step for tools and applications that deal with the detailed structure of the inflected word. It is also necessary at this step to verify that the input word is linguistically correct, i.e. it belongs to the respective language and conforms to its morphological rules. Lexical errors can be classified into the following three classes:

• Errors in word formation. These errors are related to the correct application of morphological rules. For example, it is incorrect to conjugate weak (irregular) verbs with regular verb morphological rules such as generating the imperfect form of /n-a-woSil/ instead of /na-Sil/ (we arrive), which incorrectly keeps the weak letter (۰) /w/ of the assimilated (first-weak) verb in the imperfect form.
• Errors in semantic or word choice. These errors are to some extent related to ambiguity in word senses and phonetics. For example, it is incorrect to conjugate verbs of the same root which differ in pattern by mixing up one pattern with another such as incorrectly generating the perfect tense form of the verb /ibotAEa/ (purchased) according to the pattern /ifotaEala/ instead of generating it as /bAEa/ (sold), which has the intended pattern /faEala/.
• Errors at the relationship between lexical and grammar features. These errors are related to the morpho-syntactic features of words. For example, it is incorrect to negate an Arabic verb form that occurred in the perfect using the negative particle /Lam/ (not) unless this verb is in jussive form such as /lam wajada/ (did-not find) instead of /lam ya-jid/ (does-not find).

Existing studies of lexical error analysis address two main closely related systems: spelling checkers and intelligent language tutoring systems. The purpose of most spelling checkers is neither teaching nor learning languages as they are only designed
for detecting spelling errors and suggesting possibly correct spellings (Hsieh et al. 2002). SLLs not only ask for correcting their errors by just choosing the right word from a list of alternatives but they also want to improve their language skills in order not to do the same errors over and over again. Moreover, most of the checkers are inappropriate for non-native speakers because they are mainly designed for native speakers and as such they are not suitable for detecting and correcting competence errors made by non-native speakers. For example, recent Microsoft Office’s Arabic spell checker does not detect the word /ya-qAlo-na/ (they-said) as an error. On the contrary, intelligent language tutoring systems try to overcome these problems and be more useful to SLLs by making true diagnosis of errors. Consequently, they point the learner to the right direction on how to correct their errors rather than directly providing the correct version.

In the remainder of this section, we give a brief review of the lexical error analysis and correction within both spelling checkers and intelligent language tutoring systems.

### 3.1 Lexical error analysis and correction within spelling checkers

Spell checkers are very successful at detecting and correcting basic error operations. In a written language, any misspelling can be described as a transformation from a correctly spelled word performed by one or several of the basic error operations (Ingels 1997):

- **Deletion** (e.g. /آموح/ instead of /أموح/ I-hope)
- **Insertion** (e.g. /بالم/ instead of /بالم/ interested)
- **Substitution** (e.g. /بدأ/ instead of /بدأ/ started)
- **Transposition** (e.g. /نقل/ instead of /نقل/ transfer or was transferred)

The basic error operations can be used to describe lexical errors, but they are not suitable for explaining them (Shaalan, Allam and Gomah 2003). Lexical errors can be classified according to the linguistic knowledge of a typist into the following types:

- **Typographic errors**, ‘performance errors’: The typist knows the correct spelling of the word but still makes a simple error. For example, the substitution of /الرجل/ (the-man-stood-up) for /قامل الرجل/ /qAma Alrajul/ (the-man stood-up)

---

4 Faltin (2003) defines the term term *diagnosis* as ‘identification of the cause of error’, while *correction* is ‘a thing substituted to what is wrong’. The following example illustrates the difference between diagnosis and correction. Notice that Arabic is written right to left.

- **An example of Error:** أريد أن أدرس لغة جديدة و لذلك أختار أن أدرس العربية. (I want to learn a new language so I choose to learn Arabic).

- **Error Diagnosis:** The Weak letter (و) /A/ in the hollow verb اختارت /{ixotAra/ (chose) cannot be used with the first person suffix pronoun ت /t/ because the last letter in this verb /س/ /is/ (consonant).

- **Error correction:** The correct sentence is: أريد أن أدرس لغة جديدة و لذلك أختار أن أدرس العربية. أريد أن أدرس لغة جديدة و لذلك أختار أن أدرس العربية. (where the hollow letter is curtailed.)
is a typical typographic error caused by skipping the space between two words.

- **Cognitive** and **phonetic errors**, ‘competence errors’: The typist lacks some linguistic competence (knowledge) of the language. For example, the substitution of closely pronounced letters as in /u-tiyE/ instead of /u-TiyE/ (I obey) is a typical cognitive error caused by substituting the letter ‘ت’ /t/ with the letter ‘ط’ /T/.

Another important classification of lexical errors is whether the misspelled word is either (Shaalan and Attia 2012)

- **unknown ‘isolated error’**: For example, the substitution of the misspelled word /qulo-tw/ with /qulo-tu/ (I said) is a typical isolated error caused by inserting a letter instead of a diacritic sign that gives the same pronunciation, or
- **known but refers to unintended word ‘context sensitive error’**: For example, the substitution of the word /a-tawaq~aE/ (I expect) with the word /u-waq~iE/ (I sign) is a context sensitive error where the insertion of a letter, i.e. /t/ as a second letter, changes the word meaning.

Either rule-based shallow analysis or frequencies of bigrams and trigrams can be used to detect isolated errors (Attia *et al.* 2012). Words that contained character bigrams or trigrams of low frequency were judged to be (possibly) incorrect (Daciuk 1998). This approach requires calculations of word frequencies from an available corpus. Full parsing or detailed analysis would be an ideal solution to detect context sensitive errors (Bigert 2004). The words that do not fit into the grammar are considered misplaced.

In agglutinative languages, such as Turkish, and semitic languages, including Arabic, it is impossible to base a spell checker on a word list as is often done with morphologically poor languages such as English and French, given the highly inflectional and derivational morphological nature of these languages (Shaalan *et al.* 2003). To detect misspelling errors; the program would simply try to morphologically analyze the input word; if it fails, the word is probably misspelled (Shaalan, Aref and Fahmy 2010c).

Traditional spell checkers, such as those found in present day word processors, include a spelling correction component that suggests correction as a feedback to the end user. The list of suggestions includes words present in the dictionary that are similar to the erroneous word in a certain aspect. This similarity can be measured in a number of ways, including edit distance or similarity of roots in terms of morphological categories or pronunciation (Daciuk 1998). The edit distance technique is used to measure the similarity of two strings by counting the minimal number of character insertions, deletions, or substitutions needed to transform one string into another. It has been used to only correct typographic errors.
3.2 Lexical error analysis and correction within intelligent language tutoring systems

This research addresses the development of an ILTS. One aim of this kind of systems is to enhance the teaching and learning of foreign languages (Rimrott 2005). This software can support listening, speaking, reading, and writing skills in a foreign language. Typically, ILTS provides the learner with individualized teaching and flexible feedback. It usually contains exercises for language learners whose responses are analyzed by the system in order to provide some form of detailed feedback.

Most Arabic intelligent language tutoring systems developed until now try to overcome shortcomings of general spell checkers (Shaalan et al. 2010c). They do so by incorporating morphological knowledge and non-native intuitions into their algorithms in order to be able to handle competence errors made by non-native writers. However, true diagnosis indicating the precise reason of the error is very rarely done. They still keep the behavior of spell checkers by only offering a short list of alternative words to replace the unknown erroneous words.

Shaalan (2005a; 2005b) developed an ICALL system for Arabic learners which employs deep processing and implementation of mal-rules (buggy rules) to provide error-specific feedback to morpho-syntactic errors. These specific rules make it possible to parse erroneous expressions or structures for the sake of error diagnosis. This proposed work still represents a state-of-the-art of grammatical checking within Arabic ICALL systems. The objective test method was adopted to assess learners’ knowledge. There were two main types of test items for learner’s interaction: selection type, requiring the learner to select the correct answer, and supply type, requiring the learner to write few words. In comparison with our paper, this research is simple and restricts exercises to those for which we have one and only one correct answer. While this work is admirable in many respects, error modeling and diagnosis using deep processing is limited to morpho-syntactic errors. Also, the feature set is considered simpler than the ones we used in error analysis. In our paper we investigated advanced issues that show the benefits of using deep morpho-syntactic processing.

Habash and Roth (2011) employed deep morphology to develop a general-purpose Arabic handwriting error recognition tool. A learning algorithm is used to generate a predictive model from linguistic and non-linguistic features. The handwritten text is input to an Optical Character Recognition (OCR) device and the handwritten recognition software is applied to the scanned image to convert it into text. Then the model is applied to each word in the input text in order to predict illegal words. This tool might be useful for Arabic intelligent language tutoring systems in order to guide the identification of errors in handwritten learner’s answer. In comparison to Habash and Roth (2011), our paper is about deep rule-based error analysis that uses rich linguistic features and provides adaptive feedback messages according to the learner’s level.

5 ICALL stands for Intelligent Computer-Assisted Language Learning. We use ICALL and ILTS interchangeably.
Alfaifi and Atwell (2012) have conducted a study about Arabic learner corpora. They proposed design criteria for Arabic learner corpora that include determining the participants, size of the corpora, data collection approach, selection method of representative textual data (teaching and learning material), and taxonomy of learner’s errors and their tag set. The study analysis is applied on Arabic learners from Saudi Arabia and reveals that the suggested error taxonomy is more appropriate for errors’ markup in Arabic corpora due to its accuracy and comprehensiveness. The Arabic learner corpora are not yet ready. The availability of such linguistic resource would impact not only the development of Arabic error analysis and diagnosis systems but also the performance evaluation with existing systems because we will be able to do a fair comparison based on standard training/test sets and well-defined target (i.e. errors and their corrections).

To this end, Arabic intelligent language tutoring systems are still in their initial stages compared with the work done on other languages, such as English, from which we can benefit due to the extensive research done in this field. Hsieh et al. (2002) developed an ILTS for English lexical errors. The system provides an open writing environment. They allow learners to write freely without an exercise model at the beginning. They categorized English lexical errors into three classes: compound, morphological, and spelling errors. To identify compound errors, they pre-process each compound word in the dictionary and store it in a compound word table (CWT). If the input word is in the CWT, their system recognizes it as a compound error. Morphological errors can be recognized and corrected using the reduction-inflection (R-I) algorithm. The basic idea is to reduce the input word to its base form according to several reduction rules, inflect the base form to its inflected forms, and compare the inflected forms with the input. Finally, they used minimum distance techniques to identify and correct spelling errors.

Rimrott (2003) developed a prototype program called ‘SANTY’ that provides more adequate spelling suggestions to non-native speakers of German by incorporating morphological knowledge as well as knowledge of common types of learner’s mistakes into its spell-checking algorithm. The program treats predictable verb inflection mistakes. It uses morphological analysis to determine the intended verb inflection based on the wrongly inflected input word.

The research done by Faltin (2003) and Faltin, L’haire and Ndiaye (2005) has presented an ILTS for language learners of French. Three different methods were used to retrieve correction proposals: alpha-codes, phonological re-interpretation, and ad hoc rules (pending the implementation of a morphological analyzer). Alpha-code is a simplified representation of a word, obtained by reordering the characters it is composed of, stripping the word of its diacritics, and getting rid of duplicated characters. Words sharing a single alpha-code are similar in some ways. Spelling errors that can be treated by this method are inverted letters, incorrect diacritics (including multiple incorrect diacritics) as well as some missing or superfluous characters. Phonological reinterpretation bases the similarity criteria on the sounds of the language. An expert system is used to transform the written word into its phonological representation. It is then easy to search a lexicon indexed by pronunciation for all the words that sound alike. Ad hoc rules allow their system to
find adequate correction proposals for a morphological error. This is a very limited technique, spot illegal words ending in -als or -ails. The system replaces the incorrect ending by the correct one and checks that the newly composed word is a true word before displaying it as a correction proposal. The system finally ordered the results proposed by three methods. It presents results of ad hoc rules, followed by phonological reinterpretation, showing last the alpha-code.

4 Arabic lexical errors typology

It is a major design issue for any error diagnosis system within ILTS to decide the type of errors to be diagnosed. Realistically, not every imaginable error type can be diagnosed within a single system (Faltin 2003).

Two main criteria impose themselves on the selection of error types for diagnosis. On the one hand, errors that are easy to implement, given the linguistic resources at our 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. These criteria helped us to define a set of target errors that proved relevant to SLLs of Arabic. In the remainder of this section, we explain how we identified error typology based on the above-mentioned criteria. Then we show relevant Arabic error examples for each of the selected error category.

The first step in the selection of error categories for lexical diagnosis is to study the target user needs (i.e. the SLLs of Arabic). In the literature, few linguistic studies have addressed errors made by SLLs of Arabic, including, for instance, Jassem (2000). Most of these studies followed the error analysis approach7 in identifying and classifying errors. Errors can be classified clearly and logically according to the traditional linguistic level of analysis to orthographic, phonological, morphological, syntactic, semantic, and pragmatic errors. We will focus on phonological and morphological errors because these are closely related to our research.

The common Arabic phonological and morphological errors that are handled by the proposed system are classified into three lexical error classes: Errors in word formation, errors in semantic or word choice, and errors at the relationship of lexical and grammar. Tables 2–4 detail the possible lexical errors of each of these classes.

The proposed system successfully handles single error words but restricts the handling of multiple error diagnosis. It handles multiple errors that fall under the

---

6 These examples are collected from real second language teaching environment, Malaysian Islamic secondary school learners of Arabic. The sample consisted of 54 fourth year students who were studying in the Islamic stream at the National Religious Secondary School in Kuala Lumpur. The data came from two written tasks on two topics familiar to the students, which they completed in the class (Jassem 2000).

7 Existing approaches to the analysis of a second language learner’s errors are concerned with contrastive analysis and error analysis. Contrastive analysis hypothesis compares any two or more languages phonetically, morphologically, syntactically, or lexically to predict the difficulties and errors that will occur in learning a second language. These difficulties and errors are due to differences between learner’s native language and target second language. Error analysis hypothesis studies and analyzes errors made by second language learners. This hypothesis follows six steps to identify and classify errors: data collection, error identification, error classification, error description, error explanation, and pedagogical application (Jassem 2000).
Analysis and feedback of erroneous Arabic verbs

Table 2. Word formation errors

<table>
<thead>
<tr>
<th>Error Type</th>
<th>Example</th>
<th>Correct Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Consonants errors</td>
<td>Incorrect usage of letters with the closely related pronunciation</td>
<td>(If I-learned Arabic language, I-am-able to live in Egypt).</td>
</tr>
<tr>
<td>Vowels errors</td>
<td>Making short vowel long vowel</td>
<td>(Arabic language became important these days).</td>
</tr>
<tr>
<td>Vowels errors</td>
<td>Making long vowel short vowel</td>
<td>(I-want to travel to the Middle East).</td>
</tr>
<tr>
<td>Incorrect usage of pronouns with respect to verb tense</td>
<td>(I-came to Egypt to study Arabic language).</td>
<td></td>
</tr>
<tr>
<td>Incorrect conjugation of assimilated verb in perfect tense</td>
<td>(I-called my mother to check on her).</td>
<td></td>
</tr>
<tr>
<td>Incorrect conjugation of hollow verb in perfect tense</td>
<td>(I-went to the bedroom then I-slept).</td>
<td></td>
</tr>
<tr>
<td>Incorrect conjugation of defective verb in perfect tense</td>
<td>(Mohamed escaped death by a miracle).</td>
<td></td>
</tr>
<tr>
<td>Incorrect conjugation of assimilated verb in imperfect tense</td>
<td>(We-got on the car in order to arrive the university early).</td>
<td></td>
</tr>
<tr>
<td>Incorrect conjugation of hollow verb in imperfect tense</td>
<td>(My grandmother sells rice).</td>
<td></td>
</tr>
<tr>
<td>Incorrect conjugation of defective verb in imperfect tense</td>
<td>(I-strive a lot to please my parents).</td>
<td></td>
</tr>
<tr>
<td>Incorrect conjugation of assimilated verb in imperative tense</td>
<td>(Promise by what you are able to fulfill).</td>
<td></td>
</tr>
<tr>
<td>Incorrect conjugation of hollow verb in imperative tense</td>
<td>(Always, tell the truth).</td>
<td></td>
</tr>
<tr>
<td>Incorrect conjugation of defective verb in imperative tense</td>
<td>(Always, care for your parents).</td>
<td></td>
</tr>
</tbody>
</table>

same lexical error class. An example of multiple errors that belong to the same class is when the learner writes the erroneous word /nAmo-tw/ (I slept) instead of the correct word /nim-tu/. Here the learner makes two errors that are localized to the ‘word formation’ class: incorrect conjugation of hollow verbs in perfect form...
Table 3. Semantic or word choice errors

<table>
<thead>
<tr>
<th>Error Type</th>
<th>Example</th>
<th>Correct Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Incorrect usage of root pattern</td>
<td>/a-Euwd wa &gt;a-tawa5-al AltaEoliyym binafosiy fiy AlfaSol/ (I-return to arrive teaching by myself in the class).</td>
<td>/a-Euwd wa &gt;u-wASil AltaEoliyym binafosiy fiy AlfaSol/ (I-return to continue teaching by myself in the class).</td>
</tr>
</tbody>
</table>

Table 4. Errors at the interface of lexical and grammar

<table>
<thead>
<tr>
<th>Error Type</th>
<th>Example</th>
<th>Correct Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Switching a conjugated verb with a verbal noun</td>
<td>/anA AlSalAap AlZuhor/ (I the praying the afternoon prayer).</td>
<td>/anA &gt;u-Sal&gt;iy AlZuhor/ (I pray the afternoon prayer).</td>
</tr>
<tr>
<td>Incorrect usage of verb tense</td>
<td>/&lt;iA nu-riyd &gt;an na-foham Al$aEob AlEarabiy~ min AlDaruwriy~ fahom lugathm/ (If we-want to understand the Arab people, it is necessary to understand their language).</td>
<td>/&lt;iA &gt;arAda-nA &gt;an na-foham AlSaEob AlEarabiy~ min AlDaruwriy~ fahom lugathm/ (If we-wanted to understand the Arab people, it is necessary to understand their language).</td>
</tr>
<tr>
<td>Person agreement between subject and verb</td>
<td>/*ahabo-tu &lt;ilaY Almasojid liyu-&amp;ad~iy SalAap AlZuhor/ (I went to mosque to pray-he the afternoon prayer).</td>
<td>/*ahabo-tu &lt;ilaY Almasojid li&gt;u- &amp;ad~iy SalAap AlZuhor/ (I went to mosque to pray-I the afternoon prayer).</td>
</tr>
<tr>
<td>Gender agreement between subject and verb</td>
<td>/um~iy *ahaba &lt;ilaY Alsuwq/ (My mother went-he to the market).</td>
<td>/um~iy *ahab-at &lt;ilaY Alsuwq/ (My mother went-she to the market).</td>
</tr>
<tr>
<td>Number agreement between subject and verb</td>
<td>/jad<del>iy wa jad</del>atiy {inotaqalu-wA &lt;ilaY bayot jadiyd/ (My grandfather and grandmother moved-they[plural] to a new house).</td>
<td>/jad<del>iy wa jad</del>atiy {inotaqal-A &lt;ilaY bayot jadiyd/ (My grandfather and grandmother moved-they[plural] to a new house).</td>
</tr>
<tr>
<td>Incorrect case ending of subjunctive or jussive imperfect verb</td>
<td>/jad<del>iy wa jad</del>atiy lam ya-notaqil-Ani &lt;ilaY bayot jadiyd/ (My grandfather and grandmother did not move[indicative] to a new house).</td>
<td>/jad<del>iy wa jad</del>atiy lam ya-notaqil-A &lt;ilaY bayot jadiyd/ (My grandfather and grandmother did not move[jussive] to a new house).</td>
</tr>
</tbody>
</table>

and incorrect use of long vowel instead of short vowel. In addition, the system also handles multiple errors that belong to either

(a) word formation and errors at the interface between lexical and grammar, for instance, the erroneous word أردتُ /arAdo-tu / (I wanted) instead of أريدُ /u-riyd/ (I want), or

(b) semantic and errors at the interface between lexical and grammar, for instance, the erroneous word يبيعُ /yabiyE/ (sells) instead of يُبِين /ibotAEa/ (purchased).

5 Adaptive feedback for improving learner’s linguistic knowledge

A primary objective of this research is to develop a system that provides adequate interpretations to SLL’s errors. We decided to offer feedback at the three normal
learner levels: beginner, intermediate, and advanced. The beginner learner will receive the most specific (detailed) feedback.

The proposed system output consists of the detected error type along with the issued feedback relevant to the learner’s previous performance history. For instance, a learner who has generally mastered an Arabic concept might receive a hint indicating the class of error. For the learner who generally knows the concept but still needs practice in its application, the feedback is less general such that it just points out the type of the error. For the beginner learner, the feedback is as specific as possible, and the exact source of the error is provided.

Providing adaptive feedback messages according to learner’s level follows the pedagogical principle of guided discovery learning (Heift 1998a; Heift et al. 2000). The learners are guided toward the correct answer rather than simply supplying it; thus assisting them in finding the source of an error themselves. The ability to discover the source of an error strongly correlates with the expertise of the student: the more expert the learner, the less necessary the explicit feedback (Heift 1998a). In the rest of this section we present examples of error diagnosis and adaptive feedback that illustrate the behavior of the proposed system in response to different learner’s input. For the sake of clarification, relevant Arabic error examples followed by their correct ones are given.

**Example 1:** Given the following learner input:

a. /∗a-sotayoqiZ min Alnawom mubak∼irAF/ (I-wake up early).

b. /a-sotayoqi∗ min Alnawom mubak∼irAF/ (I-wake up early).

The system detects that the learner made a phonological error ‘consonant error’ s/he switched the most closely pronounced letter (∗/) for the correct letter (/Z/); the concept of Arabic consonant letters is missing. For the advanced learner, the system issues a hint indicating a word formation mistake occurred due to phonology. For the intermediate learner, the type of the error is provided (consonant letters). For the beginner learner, the exact source of the error is also provided (usage of letters with the closely related pronunciation). Note, however, that the beginner learner is still required to decide on the correct consonant letter in the word, which is considered as a motivation to improve his/her linguistic knowledge.

**Example 2:** Given the following learner input:

a. /∗a-tanAwalo-tu AlfaTuwr mubak∼irAF/ (I-take-took my breakfast early).

b. /a-tanAwal AlfaTuwr mubak∼irAF/ (I-take my breakfast early).

The system detects that the learner made a morphological error with the word /∗a-tanAwalo-tu / (conjugation of both take and took). S/he incorrectly used two contradicted pronouns: the imperfect prefix pronoun /a- 1st pers, sg/ and the perfect suffix pronoun /tu- 1st pers, sg/ that leads to conflict in verb tense; the concept of Arabic connected pronouns is missing. For the advanced learner, the system issues a hint indicating ‘a word formation mistake occurred due to morphology’.
For the intermediate learner, the type of the error is provided (*connected pronouns*). For the beginner learner, the exact source of the error is also provided (*incorrect usage of connected pronouns with respect to verb tense*). Note, however, that the beginner learner is still required to decide on the correct tense of the verb and conjugates the verb form according to this tense (imperfect in this case).

6 The proposed architecture

The overall framework architecture of the proposed Arabic ILTS is based on a *three-tier architecture*. The *first tier* is a web browser running in the student (client) computer. The browser functions as the user interface of the tutoring module that is responsible for the presentation of materials and providing error-specific feedback message suited to the learner’s expertise level. The application programs for intelligent analysis of student input performing the student model initialization and update, and individualization reside in the *middle layer* server. In a word, this tier is responsible for receiving client requests, processing the data contained in the requests, initializing and updating the student model, and generating the adequate response based on the updated student model. Furthermore, these application components communicate directly with the backend tier that consists of databases storing information of materials and student records.

Figure 1 shows the overall framework architecture of Arabic ILTS. It consists of the following components: expert model, tutoring module, student (learner) model, item (question) banking, material generator, domain model, and graphical
user interface. In practice, this architecture might be easily extrapolated to other languages.

The expert model is an NLP component that analyzes the learner’s answer and detects possible source of errors. It includes Arabic computational tools, in particular, morphological analyzer, morphological generator, and error analyzer. Tutoring module is responsible for the initialization of the student model and issuing appropriate error-specific feedback message suited to the learner’s expertise level. It includes feedback message generator and error database. The proposed architecture keeps a record of the learner’s performance history. This information is held in the student model component. The item (question) banking, domain model, and material generator components are responsible for generating different materials or questions (tests) to the learner.

6.1 Item banking

The item banking is a database of test items such that each question in this database is linked to some learning section in the domain model. This component is used to generate different types of test items each time the learner is asking to take a test. The material generator component receives learner request and then selects test items in a random order.

There are two main types of test items for interaction with learners in Arabic ILTS: supply-type and selection-type. For supply-type items, the learner answers one or more words, such as a word in ‘fill-in-the-blank’ or a short sentence from a given sequence of roots. For selection-type items, the learner selects the answer from a pool of answers such as ‘which word is different?’ or ‘multiple-choice’. From the linguistic point of view, the type of exercises covered by Arabic ILTS can be classified as follows:

1. Supply-type questions, including the following:
   - **Dictation.** This exercise type focuses on listening skills of the learner. Learners should first listen to an Arabic sentence, then transcribe it. Next, the system diagnoses the learner answer. If it is correct, a positive feedback message is returned. Otherwise, an error-specific feedback message is provided. The learner has two additional options: either correct the error and resubmit the sentence again or peek at the correct answer.
   - **Word order.** This exercise displays a number of Arabic roots, the learner task is to rearrange these roots and conjugate them to form a grammatical Arabic sentence.
   - **Write a sentence.** In this exercise type, learners are provided with a sequence of Arabic roots. Their task is to generate a grammatical Arabic sentence using all the provided roots in the given order. The learner is warned for any extra or missing words.
   - **Transform the sentence category.** Arabic has two broad categories of sentences: nominal and verbal sentences. So, in this exercise type, learners are provided with some sentences in one category and their task is to transform these sentences into the other category.
• **Word formation practice.** This exercise displays some Arabic roots and the learner task is to conjugate these roots according to a given form.

• **Fill in blank.** The learner’s task is to complete sentences by filling in any blank with a correct verb conjugation. The learner has to use a given set of roots.

(2) **Selection-type questions, including the following:**

• **Which word is different?** This exercise displays a list of words such that all except one share the same features. The learner’s task is to identify the divergent that differs morphologically from other convergent words in some aspects, such as root, pattern, verb tense, voice, and type of weak verb.

• **Multiple choice.** The learner’s task here is to choose the correct word that matches the given features from multiple choices. The choices are different conjugations of the same word.

The structure of the item banking consists of four database relations: question, question content, answer, and concept relations.

Each question is accompanied by an associated list of concepts to test and keep a record of how well the learner has mastered them. Furthermore, each question has specific parameters that help the system in diagnosing errors. These parameters depend on the type of the test item. The set of parameters of the supply test item type is a list of feature structures (FS); each of which describes an Arabic word in the correct answer. Each FS includes the following features: correct word without diacritics, correct word with diacritics, root, pattern, type of verb, prefix string, suffix string, lexical category, tense, voice, mood, type, subject, object gender, number, and person.

For the selective test item, the set of parameters depends on the exercise type. In ‘multiple choice’ questions, the associated set of parameters is a list of feature structures that describes the intended features of each word in the answer. Each FS includes the following features: word, root, lexical category, pattern, tense, voice, mood, subject person, gender, and number. In ‘which word is different?’ question, the associated set of parameters is the intended aspect of divergence (e.g. root, pattern, lexical category, tense, voice, type of weak verb) and the value of the divergent feature for each word in the question.

### 6.2 Student model

The student model used in Arabic ILTS keeps a record of the current learning skill levels of the student (domain-specific knowledge). The student learning skill levels are represented by the frequency distribution of committed errors with regard to their

---

8 This list is selected from thirty-eight concepts defined in the system.

9 The features concerned here are related to Arabic verbs. For the purpose of handling Arabic verbs, the noun features are limited to the following: root, pattern, prefix string, suffix string, and lexical category.

10 This feature shows the type of the word either conjunction or question.
Analysis and feedback of erroneous Arabic verbs

Table 5. An extraction of the student model database table

<table>
<thead>
<tr>
<th>Concept Id</th>
<th>Error name</th>
<th>Error parameters</th>
<th>Error Knowledge Id</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>make short long</td>
<td>[short, long]</td>
<td>con2_error1</td>
</tr>
<tr>
<td>2</td>
<td>make long short</td>
<td>[long, short]</td>
<td>con2_error2</td>
</tr>
</tbody>
</table>

Table 6. An extraction of the student skill database table

<table>
<thead>
<tr>
<th>Student ID</th>
<th>Error knowledge Id</th>
<th>Error ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>2012123</td>
<td>con2_error1</td>
<td>[3,4]</td>
</tr>
</tbody>
</table>

correction. Therefore, we calculate the individual learning skill level in terms of the frequency of an error that the student has made while fulfilling its related concept. For example, the learner may be a beginner with regard to ‘incorrect use of root pattern’ error but advanced with ‘make a long vowel short’ error. This knowledge is represented by a perturbation error model. In this model, we assume the presence of one or more misconceptions for each concept that are related to teaching of Arabic weak verbs. By this way, the student model knowledge is represented by the union of a subset of the domain knowledge (Arabic concepts of weak verbs) and another subset of the misconception set. In the proposed Arabic ILTS, the student model is a database of error records. Table 5 shows an extraction of the student model database table which defines that the vowel letters concept (ID = 2) has two associated misconceptions: make a short vowel long and vice versa. The complete student model database lists all Arabic concepts and their associated misconceptions defined in the domain model.

Another database table showing student skill is used to express current skill of each student. The frequency of the error is expressed by a number pair, i.e. (EFC, TC), where EFC is the Error Frequency Counter variable that indicates how many times the student has made this error, and the TC variable is the total counter that indicates the total number of times for which the student has met this concept. As the frequency of the error becomes higher, the student requires detailed instruction. For example, Table 6 shows that the student (ID = 2012123) has made the error that replaces the short vowel with the long one three times while s/he was practicing the vowel letters concept four times. Therefore, for each error of certain concept, the student model keeps a proportion of its frequency to the total times the concept has been met such that the ratio falls in the range of one of the three learning skill levels. The learning skill level determines the level of details of the feedback. We decided to determine the student learning skill level as follows:

11 In Arabic ILTS, a specified list of parameters for each error is defined. For example, the ‘make short vowel long one’ error has a parameter list identifies what vowel is being the other [short, long].

12 The system has 107 student skill entries for each registered student in the system.
• **Beginner**: Frequency of the error > 0.50.
• **Intermediate**: 0.20 < frequency of the error ≤ 0.50.
• **Advanced**: 0 ≤ frequency of the error ≤ 0.20.

Hence, in Table 6 the learning skill level of the student 2012123 with regard to *make a short vowel long* error is 75 percent because he did this error three times out of four. This is interpreted as, the student’s skill is still at the beginner’s learning level.

### 6.3 Expert model

This component is responsible for analyzing learner’s answer, detecting possible error types, and accumulating the frequency counters. The expert model performs NLP tasks through Arabic computational tools: *morphological analyzer*, *morphological generator*, and *error analyzer*. In particular, the expert model retrieves from the Item Bank the intended features of each word in the question (i.e. question’s parameters), concepts associated with each question, and the test item type. Then, according to the test item type, it proceeds with the appropriate procedure for analyzing the learner’s answer.

As the correct answer of the selective test item type is predetermined, the processing does not need deep Arabic NLP. However, it is needed for the supply test item type in order to get all possible analyses of the learner’s answer and then to select the most appropriate one according to the instruction learner level and difficulty of Arabic concepts\(^\text{13}\) (cf. Shaalan *et al.* 2010a; 2010b). For example, the passive Arabic verb might have the same orthographic form as the active one. However, it is doubtful that a beginner learner of Arabic would use a passive voice verb instead of an active voice. So it is obvious in this case that passive voice is a rare construction. Hence, we allow some *prioritized conditions* in order to select the most preferred word analysis, such as ‘if the question goal is to test passive voice then the system selects passive voice analysis otherwise, it selects the active voice analysis’.

#### 6.3.1 Analysis of answers to the selective test item

This section presents an explanation of how Arabic ILTS analyzes learner’s answer in case of selective test item questions. Consider the following example as well as the detailed steps of the analysis algorithm that is relevant to this example.

**Example 3**: Choose the correct answer from the word given in parentheses:

I ... rain today (expect - sign - expected)

If the learner has made a choice to select the wrong word /u-waq~iE/ (I-sign) instead of the correct word /a-tawaq~aE/ (I-expect), then s/he has

\(^{13}\) Due to the complexity and importance of this step, a complete illustration of how the analyzer selects the appropriate analysis is presented in details in the next section.
incorrectly used the imperfect tense of the root /w-q-E/ with the pattern /tafaE-al/ instead of the intended pattern /afoEal/.

The system applies the following steps to detect this error:

**Step 1. Increment the TC variable for all relevant concepts in the answer.**

For example, the word /a-tawaq∼aE/ (I-expect) has two associated concepts: *verb pattern* and *usage of imperfect active verb*. The *verb pattern* concept has two possible associated errors: *incorrect use of root pattern* and *confusion in choosing verb pattern*. Assume that the current frequency counters of these errors are [0, 1] [1, 2], respectively. These are updated by incrementing the variable TC such that the frequency counter pair became [0, 2], [1, 3], respectively.

**Step 2. Match the features in the learner’s answer with the retrieved features of the correct answer.** If a match is achieved, produce a positive feedback and exit.

Since there is a mismatch between the input (/u-waq∼iE/) and the correct answer (/a-tawaq∼aE/), the processing will continue to the next step.

**Step 3. Perform shallow error analysis to detect the possible source of error in the learner’s answer and accumulate the relevant error frequency counter variables.** In the next section we provide more detailed error analysis and explain the used techniques.

The error analyzer detects that the learner has made ‘**incorrect use of root pattern**’. As a result, the error type ‘**verb pattern**’ and the parameters (the erroneous pattern and the correct pattern, i.e. [‘ Incorrect’, ‘Incorrect’]) are passed to the tutoring module in order to issue appropriate feedback suited to the learning level.

Moreover, the EFC variable is incremented for the detected errors. So in this case the variable EFC of the ‘**incorrect use of root pattern**’ error is incremented and the frequency counter pair became [1, 2].

### 6.3.2 Analysis of answers to the supply test item

This section presents an explanation of how Arabic ILTS analyzes learner’s answer in case of supply test item questions. Consider the following example as well as the detailed steps of the analysis algorithm that is relevant to this example.

**Example 4:** Write a sentence using the following roots and follow the given right-to-left order:

/\r-w-d, >an, $ -k-r, li>an∼aki, z-w-r, n-z-l, fiy, E-y-d/.

Note that question head includes five roots, two particles, and one preposition.

Assume that the following is a possible learner’s answer:

/\r-w-d, >an, $ -k-r, li>an∼aki ta-zuri-yna manoz-iliy fiy AlEiyd / (I-want to thank-you because-you visit my-house in the-feast)

While the correct answer is:
In this case, the learner made two errors with regard to the erroneous word /ta-zuri-yna/ (you-(f, sg) visit):

1. Incorrect conjugation of ‘imperfect active verb in indicative mood’ instead of ‘perfect active verb’.
2. Incorrect use of the diacritic Damma (/u/) as a short vowel instead of the long vowel (’/w/).

The following linguistic knowledge is found relevant to the illustration of the analysis of the learner’s answer:

- The feature structure of the inflected verb form /zuro-ti/ (you-(f, sg) visited), which is answered incorrectly by the learner.
- Feature structure:
  
  Correct answer:
  Correct answer with diacritics: زُرْت
  Root: ®
  Lexical category: verb
  Pattern: فعل
  Verb type: hollow
  Prefix: ‘
  Suffix: ت
  Tense: perfect
  Voice: active
  Mood: ‘
  Type: ‘
  Subject person: 2
  Subject num: sg
  Subject gender: f
  Object person: ‘
  Object num: ‘
  Object gender: ‘

- Seven associated concepts: conjugation hollow verb in perfect tense active voice, subject verb agreement, and usage of perfect active verb in addition to four concepts commonly associated with any Arabic words (i.e. consonant letters, vowel letters, lexical category, and pattern).

The system applies the following steps to detect these errors:

**Step 1. Increment the TC variable for all relevant concepts in the answer.**

**Step 2. Match the features in the learner’s answer with the retrieved features of the correct answer.**

- If we get an exact match, then produce a positive feedback and exit. Else if we get a partial match, then delete only the exact matches from both answers.

Applying this step on the above example would leave us with one word on each side. The word that remains in the correct answer is /zuro-ti/ (visited (f, sg)), whereas the word that remains in the learner’s answer is /ta-zuri-yna/ (you-(f, sg) visit).
Step 3. Verify whether the learner has overlooked any word, inserted extra words, or changed the order of any word from the input. It uses constraint relaxation and edit distance techniques to perform this function. This step may produce missing word, extra word, and word disorder warnings. In the next section, we provide more detailed error analysis and explain the used techniques.

Since no such warnings exist, the processing will proceed to the next step.

Step 4. Perform morphological analysis to each word in the remaining learner’s answer and get all possible analyses, i.e. creating one-to-many analyses. Then, for each analyzed word, send the error analyzer module only the analysis that shares its root with a root in the correct answer.

Applying the morphological analysis on the remaining learner’s answer 

will not succeed as the word is ill-formed. Thus, an empty list is sent to the error analyzer module, indicating no possible analysis could be reached.

Step 5. Perform error analysis to each of the remaining words in the learner’s answer. As words might be ill-formed, it extracts their main features.

Applying error analysis on the remaining words in the learner’s answer, it produces three possible feature structures of the erroneous word 

(1) The stem is /zuri/. The main features are that the word is a second person feminine singular imperfect verb active voice in indicative form.

(2) The stem is /taziri/. The main features are that the word is a second person feminine singular imperative verb with first person plural object.

(3) The stem is /taziri/. The main features are that the word is a second person feminine singular imperative verb with first person singular object.

Step 6. Perform morphological generation using the extracted feature structures of the learner’s answer and their corresponding roots in the correct answer to get candidate-inflected stem forms.

Applying the morphological generator on the root /z-w-r/ for all three feature structures, we get their corresponding candidate stem forms. In the current example the three feature structures generate the same stem /zuwri/

Step 7. Perform error analysis using the edit distance technique on two kinds of stems, i.e. analyzed stem from the ill-formed input (from step 5), and its corresponding stem that is generated in reverse (from step 6), in order to detect all possible error types and update the student model accordingly.

In the current example, the error analyzer discards both the second and third analyses of step 5 since the generated stem /zuwri/ and the analyzed stem /taziri/ do not match. However, there is a match between the first analysis

14 The Arabic ILTS has used the online Xerox morphological analyzer accessed through http://www.xrce.xerox.com/. Similar to most morphological analyzers, this tool can only analyze well-formed Arabic word.
/zuri/ and the generated stem /zuwri/ when the missing weak letter\textsuperscript{15} ‘/waw/ is restored. Consequently, the error analyzer successfully detects the two errors made by the learner for the erroneous input word /ta-zuri-yna/ (you-(f, sg) visit) – (incorrect conjugation of ‘imperfect active verb in indicative mood’ instead of ‘perfect active verb’, and incorrect use of the diacritic (\textsuperscript{15}ضمية) /u/ as the short vowel Damma instead of the long vowel (\textsuperscript{15}ضمية) /w/). More illustrations regarding the techniques used during the error analysis are presented in the next section.

Note that while it is desirable to construct a system capable of detecting and accurately explaining all errors, it does not follow that the system should display each and every error detected. The proposed Arabic ILTS filters all detected errors and selects one relevant error in order not to overwhelm the student with many feedback messages. In the current example, the Arabic ILTS selects the incorrect conjugation of ‘imperfect active verb in indicative mood’ instead of ‘perfect active verb’ as a source of error because the ‘incorrect use of the short vowel Damma (\textsuperscript{15}ضمية) /u/’ instead of the ‘long vowel (\textsuperscript{15}ضمية) /w/’ depends on the selected error.\textsuperscript{16} Finally, the expert model sends the verb tense as the error type to the tutoring module along with the following list of parameters: [(tense: ‘perfect’, voice: ‘active’, mood: ‘’)] (tense: ‘imperfect’, voice: ‘active’, mood: ‘indicative’)]. The tutoring module in turn issues appropriate feedback suited to the learning level. The student model is finally updated by incrementing the EFC variable for the two diagnosed errors, ‘incorrect use of imperfect active verb in indicative mood’ and ‘make a long vowel short’, which belong to the concepts of vowel letters and usage of perfect active verb, respectively. A complete explanation of the tutoring module is given in the rest of this section.

6.4 Tutoring module

This module is responsible for initializing the student model. When a new learner is registered, all the frequency counter pairs are set to [0, 0], which assigns advanced level to the learner. Further practice might step him/her down to either beginner or intermediate learning level.

The feedback system is also responsible for issuing appropriate error-specific feedback message suited to the learner’s expertise. There exist three levels of errors that correspond to the following learning skill levels kept in error database relations: error class, error type, and source of errors. An example of the error class entry is the ‘word formation due to phonology’ (Arabic translation ترتيب الكلمة لسبب صوتي) lexical error. An example of the error type entry related to this class is ‘vowel letter’ (Arabic translation حرف صوتي).

\textsuperscript{15} The weak letter /waw/ is also a long vowel letter such that learner usually mixes it with the short vowel Damma. Short vowels in modern standard Arabic are optional. A weak letter can be removed by Arabic conjugation. So, due to this complexity in Arabic, a learner might erroneously applies a wrong conjugation that removes this weak letter and assumes that the short vowel Damma replaces it.

\textsuperscript{16} For more details about how the system filters errors, the reader is referred to the next section.
Examples of the source of errors that belong to the same error type are: ‘make a short vowel long’ (Arabic translation: طولة الحروف القصيرة) and ‘make a long vowel short’ (Arabic translation: تقصير الحروف الطويلة), which belong to the ‘vowel letter’ error type.

The feedback message generator module receives a number between 1 and 3, indicating the learning skill level according to this error and the error type along with its parameters.

The general procedure for feedback generation works as follows: For the advanced learning level, the feedback is most general, providing a hint to the class of error. For the intermediate learning level, the feedback is more detailed, providing additional information on the type of error. For the beginner, the feedback is the most precise one. It not only pinpoints the class and type of the error but also refers to the source of the error. For example, if the learner writes the word “زورت” /zuro-tu/ (I-visited) instead of “زرت” /zir-tu/ (I-visited), the source of the error here is ‘make a short vowel long’. The expert model detects this error and sends to the ‘vowel letter’ error type of tutoring module with the following parameters: ['short', 'long']. The feedback generation module checks the learner skill level. For advanced learner, it issues the following error class message: ‘(word formation due to phonology). For intermediate learner, it issues the following error class message: ‘خطأ في تراكيب الكلمة لسبب صوتي’ (vowel letter). For beginner learner, it issues the following source of error: ‘(make a short vowel long).

It is worth noting that there is a specific procedure for the feedback message generation of ‘which word is different?’ test item question. It works as follows: For the advanced learning level, the feedback is most general, providing a hint to the type of the error (i.e. the divergent aspect such as root, pattern, lexical category, tense, voice, and the type of weak verb). For the intermediate learner, the feedback is more detailed, providing additional information on the value of divergent feature for learner answer. For the beginner learning level, the feedback is the most precise one. It not only pinpoints the type of the error but also refers to the exact source of the error (the value of the divergent feature for both correct and learner answer). For example, consider that the divergence aspect is ‘type of weak verb’ (نوع الفعل المتعت) when all choices in the question are defective perfect verbs except one, which is a hollow perfect verb. Then, by selecting a defective verb, the source of error becomes ‘خاطئ في نوع الفعل المتعت’ (confusion between defective and hollow verbs). The expert model should detect this error and send the tutoring module the error type ‘type of weak verb’ with the parameters: defective and hollow. The feedback generation module checks the learner skill level. For advanced learner, it looks up the error type database and issues the corresponding error message: ‘(wrong selection: the error is in the type of a weak verb). In the case of intermediate learner, it issues a combination of the error type and the value of the divergent feature error class message: ‘(wrong selection: the error is in the type of a weak verb. This word is a defective verb). Finally, for beginner learner it issues the following source of error: ‘خاطئ في نوع الفعل المتعت’ (Wrong
selection: The error is in the type of the weak verb. This word is a defective verb, while the correct answer is a hollow verb).

7 Error diagnosis: a closer look

The previous section presents a sketch of main components of the proposed Arabic ILTS as a whole. In this section we describe in detail the automatic error analysis component that is a part of the expert model. We used the benefits of finite-state automata for representing context-sensitive linguistic information that is required for implementing constraint relaxation and edit distance techniques.

7.1 Error diagnosis of selective test item

In the selective test type, error diagnosis is straightforward. It is guided by the feature structures describing the main properties of each word in the answer. For example, the following is the FS of the well-formed word أقول / >-a-quwl/ (I-say):

```
Word: أقول
Root: ـ-وـ-ل
Lexical category: verb
Pattern: فعل
Tense: imperfect
Voice: active
Mood: indicative
Subject person: 1
Subject num: sg
Subject gender: neutral
```

The comparison of feature-value pairs between the correct FS and the erroneous FS leads to identifying the error type. The tutoring module receives as input the error type, error parameters, and learning skill level of this error, and produces as output the appropriate feedback. For example, consider the following ‘multiple choice’ question, which asks the student to select a correct answer from a given list of words.

Example 5: Choose the correct answer from the words between parentheses:

I always ... the truth. (tell – told – tells)

The correct answer is أقول / >-a-quwl/ (I-tell). If the learner answers قال قل /qAla/ (told), a perfect verb derived from the root قـ-وـ-ل /q-w-l/ and pattern فعل /faEala/.

17 Lexical errors that arose in this type of questions are either semantic or at the border between lexical and grammar, such as ‘verb pattern’, ‘lexical category’, ‘verb tense’, and ‘subject–verb agreement’.
then this answer will be diagnosed as a *verb tense* error. The parameters that indicate the mismatches are: *imperfect active verb in indicative mood* form in the correct answer and *perfect active verb* in the learner’s answer. The tutoring module issues the feedback message that is suitable for the error learning skill level:

- **Advanced learner**: ‘خطأ صرفي أو نحوي في سياق الكلام’ (morphological or syntactic context sensitive error),
- **Intermediate learner**: ‘خطأ في زمن الفعل’ (verb tense error), or
- **Beginner learner**: ‘استخدام خاطئ للفعل ماضي مبني للتعلم بدلاً من فعل مضارع مرفوع مبني للمعلوم’ (incorrect use of perfect active verb instead of imperfect active verb in the indicative form).

In the ‘*which word is different?*’ question, the parameters are the aspect of divergence (such as root, pattern, lexical category, tense, voice, type of weak verb), and the possible values. For example, consider the following question, which asks the student to select the word that differs from the other words.

Choose the divergent word:

- Summon
- Strived
- Threw
- told

The correct answer is the divergent word *قال* /qAla/ (told). All convergent words are *defective* (a weak final radical) *perfect verbs*, whereas the divergent word is a *hollow* (a weak middle radical) *perfect verb*. In this case, the aspect of divergence, the *type of weak verb*, is the error type. The parameters that indicate the mismatches are: *hollow verb* in the correct answer and *defective verb* in the learner’s answer. The tutoring module issues the feedback message that is suitable for the error learning skill level:

- **Advanced learner**: ‘اختيار خاطئ: الخطأ في نوع الفعل المعطل’ (wrong selection: the error is in the type of a weak verb).
- **Intermediate learner**: ‘اختيار خاطئ: الخطأ في نوع الفعل المعطل’ (wrong selection: the error is in the type of a weak verb. This word is a defective verb), or
- **Beginner learner**: ‘نوع الفعل المعطل هذه الكلمة هي فعل ناكس بينما الإجابة الصحيحة هي فعل أجوف’ (Wrong selection: The error is in the type of a weak verb. This word is a defective verb, while the correct answer is a hollow verb).
7.2 Error diagnosis of a supply test item

Error diagnosis of a supply test item – where the learner is asked to write a few words – is more sophisticated than a selective type, since the student is allowed to be more creative. So it is necessary in the proposed Arabic ILTS to incorporate morphological knowledge and non-native intuitions into its error diagnosis algorithm in order to be able to handle competence errors made by non-native learners. The preconditions for error analysis are concerned with verifying whether the learner has overlooked, inserted, or modified the required order of any word in the input. The learner should correct the input and make sure that the answer is free of such warnings. The current system is organized in such a way that only one warning at a time is reported to the learner. Example 6 shows answers that cover these errors. The rest of this section gives more insights on the diagnosis steps followed by the system.

Example 6: Write a sentence using the following roots and follow the given right-to-left order.
- /d-w-m/-
- /H-q/-
- /q-w-l/-

There are three roots. Assume the following five possible answers.

a) /qAltw AlHaq dA\{imAF/ (I always tell the-truth).
b) /a-quwl dA\{imAF/ (I always told).
c) /a-quwl AlHaq dA\{imAF muxoliSF/ (I always tell the-truth faithfully").
d) dA\{imAF qAlo-tw AlHaq~ (always, I-told* the-truth)
e) qAlo-tw AlHaq~ dA\{imAF/ (I always told* the-truth).

Answer (a) is a correct answer. In answer (b), the second root (/H-q/) is missing. In answer (c), the learner has added an extra word (/muxoliSF/) at the end of the sentence. In answer (d), the learner has not followed the required word order. In answer (e), the learner input has met the conditions of not missing a given word, adding extra word(s), and not violating the word order. However, s/he has made lexical errors in the first word /qAltw/ (I-told). The lexical error analysis performs a comparison between features of words in the correct answer and features of words in the input, and concludes that in answer (e) the learner has made three lexical errors18 with the word /qAlo-tw/ (I-told). The following section uses this answer, as a working example, to show how the word and lexical error analyzers are able to diagnose these errors. It is difficult to have one single example that demonstrates the functionality of each diagnosis step and how it causes the solution structure to evolve. So, for completeness, we tried our best to explain each step in the error diagnosis algorithm using answers in Example 6. Besides, we give additional examples and explanations as necessary.

---

18 The three errors are as follows: (1) incorrect conjugation of perfect active verb /qAla/ (told) instead of imperfect active verb in indicative form /a-quwl/ (I tell)), (2) incorrect conjugation of hollow verbs in perfect tense (middle letter /A/ should be removed), and (3) incorrect use of long vowel (last letter /w/) instead of short vowel (case ending /u/).
7.2.1 Identification of base forms

The result of the match between the features in the learner’s answer with the retrieved features of the correct answer is exact matches and mismatches. We are interested in analyzing the latter, so we delete the exact matches from the input. Then the word analyzer applies constraint relaxation and edit distance techniques on each word in the learner’s answer and produces all possible base word forms. The base word form is defined as a normalized root form, i.e. the root after removing any weak or hamza letters, which facilitates the comparison among different verb conjugations that are derived from the same root without doing extra processing for recognizing the lexographic change in such irregular forms. For example, the base form of both /أَلْقِ/ (I-tell) and /قَالَ/ (told) is /قَلَ/. The base form is used to verify whether the student overlooked a given word or introduced a new word in his/her answer.

The word analyzer also suggests initial error detection hypotheses that will be confirmed later on by the lexical error analyzer. The constraint relaxation and edit distance techniques split the erroneous word into three segments: prefix + stem + suffix. With the constraint relaxation technique the partial structures can combine only if certain constraints imposed by the language are met. When these constraints are relaxed, an attachment is allowed even if the constraint is not satisfied. The relaxed constraint must be marked on the structure such that the type and the position of the detected error can be confirmed later on. As Arabic is a morphologically rich language, various constraints should be met to form a well-formed inflected word, such as imposing certain connected pronouns on specific Arabic verb tense and imposing certain affixes or clitics (e.g. conjunction or pronoun) on specific conjugated verbs. In the proposed Arabic ILTS these two example constraints can be relaxed to allow for error diagnosis.

The following steps are used to get base word forms from all words in the learner input:

1. Extract a list of all possible suffixes.
2. Filter the suffix list to exclude irrelevant suffixes using heuristic rules.
3. Extract a list of all possible prefixes.
5. Extract a list of all possible stems.
7. Normalize the stem to get base word forms.

In the following we show the application of each of these steps on the word /قَالَ/ (I-told), the remaining word from answer (e) of Example 6, to get the base word forms.

**Step 1. Extract a list of all possible suffixes**

The system uses regular expression to describe the list of affixes.\(^{19}\) The regular expressions are represented as deterministic finite-state automata. The suffix list is

\(^{19}\) The list of affixes described in Buckwalter (2002) is used.
expressed in reverse order to efficiently parse the input left to right, i.e. the suffix characters sequence is matched from the leftmost (suffix) positions of the words. Figure 2 illustrates a finite-state automata representation of four suffixes /Teh, Alef, Alef-Teh, Waw-Alef/.

To extract the suffix list, the system parses the input word against the suffix automata. The parsing begins at the end of the input word and works backward. The system relaxes the usage of certain affixes constraint by using the three-way-match method to compare two strings (Elmi and Evens 1998): suffix of learner input with an Arabic proper suffix. This method assumes that when a character at location \( n \) in the first string does not match the character at location \( m \) in the second string, we have an error\(^{20}\) 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 \( m \)), where \( n \) represents a position at the input string and \( m \) represents a position (state) at the finite-state automata. Initially, \( n = 1 \) to point the last letter in the input string and \( m = 0 \) to point the letter at the initial state (last suffix letter) in the finite-state automata. The three-way-match comparison and the order of the comparison are shown in Figure 3.

For example, given the finite-state automata shown in Figure 2 and the input word /qAlo-tw/ (I-told), the word analyzer proceeds as follows: (1) match the last letter \( (n = 1) /w/ \) of the input word with the Arabic proper suffix \( /t/ \) that occurs at the end \( (m = 1) \) of Arabic verbs. The match fails. So it tries to match again with the letter but last \( (n + 1 = 2) \) of the input word, i.e. letter \( /t/ \), which succeeds. This process considers the letter \( /t/ \) as a possible suffix and the letter \( /w/ \) as an

\[^{20}\text{The error may be either missing, inserted, or substituted character.}\]
extra letter occurring at the end of the input word. Similarly, the match exhaustively proceeds with other Arabic proper suffixes yielding ten possible solutions as shown in Table 7.

In practice, the use of constraint relaxation technique in the analysis leads to over-generation. In order to resolve this issue and to identify the analyses which best reflect the learner's intention, we applied a set of heuristic rules to eliminate highly implausible analysis. For example, SLLs of Arabic find it difficult to differentiate vowel signs from genuine characters, such as the confusion about the use of the short vowel Damma (اُم) /u/ versus the long vowel letter /w/. Therefore, the system restricts the inserted or missing characters to only weak letters. Moreover, the system restricts the converted character to be a character from a local set of similarly pronounced characters. The closely related pronounced characters are grouped into the following sets:

\{
{ش, 'د', 'ت', 'ح', 'ث', 'ص', 'ص'},
{س', 'س', 'ص', 'ص'},
{ق', 'ق'},
{ع', 'ع'},
{ة', 'ة'},
{ي', 'ي'}.
\}

**Step 2. Filter suffix list using heuristic rules**

The suffix list could be reduced tangibly in a number ways taking into consideration the set of error categories handled by the system. For example, the list of possible suffixes in Table 7 could be minimized to the following five solutions:

- Suffix solution (1): NULL suffix.
- Suffix solution (4): First person singular perfect verb suffix with extra Waw.
- Suffix solution (8): Second person male plural imperative verb suffix with missing Alef.
- Suffix solution (9): Male plural imperfect verb suffix with missing Alef.
- Suffix solution (10): Third person male plural perfect verb suffix with missing Alef.

Consequently, five (half of) solutions were removed. There are two reasons for these decisions. Currently, the system does not handle errors related to Arabic nouns, hence led to ignoring the second and the seventh solutions. Nevertheless, the other three solutions (third, fifth, and sixth) are discarded since their end case mismatches the extra character /w/. This error type is the misconception of making a short vowel long vowel.

**Step 3. Extract a list of all possible prefixes**

Extracting the prefix list is similar to extracting the suffix list except that the order of the match process begins at the first letter and upwards. For example, applying this step on the word قلأ (I-told), we get only a null prefix in the prefix list as shown in Table 8.

**Step 4. Filter prefix list using heuristic rules**

The prefix list could be minimized according to the set of error categories handled by the system. For example, the system does not handle errors related to Arabic nouns. So any solution related to nouns is just ignored. Applying this step on the word قلأ (I-told) does not filter the result.

**Step 5. Extract all possible stems**
In order to extract a possible stem, the system tries to get every possible substring from the input word that remains after relaxing morphological constraints. In this case the constraint under consideration is ‘Imposing certain pronouns on specific verb
Table 8. The list of possible prefixes of the input word /qAlo-tw/ (I-told)

<table>
<thead>
<tr>
<th>Solution description</th>
<th>Solution structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Null prefix</td>
<td>Prefix string: “</td>
</tr>
<tr>
<td></td>
<td>Prefix id: Pref-0</td>
</tr>
<tr>
<td></td>
<td>Prefix FS: [lexical_category: neutral]</td>
</tr>
<tr>
<td></td>
<td>Error indication: []</td>
</tr>
</tbody>
</table>

*tense* (i.e. although the tense feature of the verb does not agree with its prefix and/or suffix, this solution is still counted). Otherwise, i.e. there is no such constraint to be relaxed, the system tries to verify the compatibility rules between a prefix and a suffix from a predefined list of prefixes and suffixes, discarding any disagreement due to incompatible combination.

The output of this step yields four solutions as shown in Table 9. Each solution consists of five elements constituting the breakdown of the word into three segments: *prefix, stem, and suffix*, an *FS* that describes the analyzed word, and an initial *error hypothesis*. The affix is represented by three components: the *affix itself*, an *ID*, and the *affix’s FS*. The stem is represented by the actual sequence of Arabic script. The error hypothesis is represented by a structure that indicates the operation (insert, delete, or convert) required to relax the affix and whose elements are the actual characters, the position where the operation should take place, and the affix tag.

Note that in the working example, the discarded incompatible prefix and suffix combination is the solution ‘Null prefixed masculine plural imperfect verb with deleted Alef in the suffix’ (i.e. the ninth suffix solution in Table 7) because in Arabic morphology it is invalid to use a combination between a null prefix and the imperfect suffix ‘َا’ (Waw and Alef) and there is no matter of relaxing this incompatibility issue. This suffix ‘َا’ can only be used with either the prefix ‘َه’ (Yeh) or ‘َث’ (Teh).

**Step 6. Group identical stems together**

The solution list may include similar stem strings. For the sake of efficiency, each group of similar stem strings is combined together to form an element containing the common stem string and a group of pointers to the corresponding elements in the solution list. After applying this step to the four solutions in Table 9, a list of three elements is formed: `{(، 1), (، 2), (، 3, 4)}).

**Step 7. Normalize the stem to get the base word forms**

To get all possible base word forms from each stem, the list of Arabic verb patterns is expressed as deterministic finite-state automata. Figure 4 illustrates a finite-state automata of the patterns /faEala, fAEala, tafAEala/.

We differentiate between two types of letters forming the pattern (template): *radical* and *variant* ones. For example, the pattern /fEala/ has three radicals, while the pattern /tafoE~al/ has one variant letter /ِ/ and three radical letters.

21 The compatibility rules between prefixes and suffixes of the Buckwalter (2002) Arabic morphological analyzer is used. For example, the null prefix ‘Pref-0’ is compatible with the third person feminine singular perfect verb suffix ‘PVSuff-at’.
Table 9. The list of word analyses of the input word /qAlo-tw/ (I-told)

<table>
<thead>
<tr>
<th>Solution Description</th>
<th>Solution Structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>1- Null affix</td>
<td>Prefix: ('', Pref-0, [lexical_category: neutral])</td>
</tr>
<tr>
<td></td>
<td>Stem string:</td>
</tr>
<tr>
<td></td>
<td>Suffix: ('', Suff-0, [lexical_category: neutral])</td>
</tr>
<tr>
<td></td>
<td>Word FS: [lexical_category: neutral]22</td>
</tr>
<tr>
<td></td>
<td>Error indication: []</td>
</tr>
<tr>
<td>2- Null prefix with first person singular perfect verb in active voice with extra Waw in the suffix</td>
<td>Prefix: ('', Pref-0, [lexical_category: neutral])</td>
</tr>
<tr>
<td></td>
<td>Stem string:</td>
</tr>
<tr>
<td></td>
<td>Error indication: [insert(4,5,affix)]</td>
</tr>
<tr>
<td>3- Null prefix with second person masculine plural imperative verb with deleted Alef in the suffix</td>
<td>Prefix: ('', Pref-0, [lexical_category: neutral])</td>
</tr>
<tr>
<td></td>
<td>Stem string:</td>
</tr>
<tr>
<td></td>
<td>Error indication: [delete(1,6,affix)]</td>
</tr>
<tr>
<td>4- Null prefix with third person masculine plural perfect verb in active voice with deleted Alef in the suffix</td>
<td>Prefix: ('', Pref-0, [lexical_category: neutral])</td>
</tr>
<tr>
<td></td>
<td>Stem string:</td>
</tr>
<tr>
<td></td>
<td>Error indication: [delete(1,6,affix)]</td>
</tr>
</tbody>
</table>

Fig. 4. A finite-state automata for four Arabic verb patterns: /faEala, fAEala, tafO~al, tafAEala/. The letter inside a square is generic.

The system uses the three-way-match method to parse each stem in the solution against the actual verb pattern and relaxes any missing or transformed similar letters. If the parse succeeds, it deletes from the resultant stem any weak or hamza letters, producing the base word form.

22 This feature structure represents the third person masculine singular perfect verb in active voice.
The parsing is applied to the following stem list: \{([لَا, [2]), ([لَا، [3, 4])}\}. The parsing of the first solution is not applicable because there is no Arabic pattern that might match. The parsing of the second solution yields intermediate results \{قَلَّ, قَلَّ\}, which after removing the weak letter from the first stem become the same base form \{قَلَّ\}. The parsing of the third solution yields the intermediate results \{قَلَّت, قَلَّت\}, which after removing the weak letter would become the base form \{قَلَّت\}. Hence, the final solution of word analysis yields the base forms \{([لَا، [2]), ([لَا، [3, 4])\}, which correspond to the following analyses:

- Null prefix, first person singular perfect verb in active voice, and extra suffix Waw.
- Null prefix, second person masculine plural imperative verb, and deleted suffix Alef.
- Null prefix, third person masculine plural perfect verb in active voice, and deleted suffix Alef.

### 7.2.2 Verify preconditions

The verification of preconditions is a pipeline process that checks whether the learner’s answer exactly meets the question requirement of including every given word and following the same order. A warning is issued in case of violating any of these requirements and the learner has to satisfy these preconditions before the error analyzer delves into deep analysis. The comparison involves features of the base forms of words in the learner answer versus features of the base forms of retrieved roots.

*Extra Word Checker:* The extra word checking is applied on both base forms, and the system reports an error once an extraneous word is found in the learner input. As an example, consider answer (c) in Example 6; comparing base forms from both sides results in the following extra word مخلص (faithfully) in the answer.

*Word Order Checker:* The word order checking is applied on parallel to the sequence of base forms of the learner answer and retrieved roots. The system reports an error once a difference is found. As an example, consider answer (d) in Example 6, comparing base forms in parallel results in a difference in the first word.

*Missing Word Checker:* The precondition of not missing a word is verified by matching each base stem form produced from the word analysis with the root base form. Any overlooked root is reported. As an example, consider answer (b) in Example 6, where the second base form is missing. An interesting result from the missing word checking is that any irrelevant base form is removed from the possible words in the analyses of the answer. For the sake of clarification, the following are found relevant to the working example:

- The base form of the root جَرَى /q-w-l/ that is given in the question is \{قَلَّ\}/q-1/.
- The base forms of the input that result from the answer analysis are \{([قَلَّ, [2]), ([قَلَّت, [3, 4])\}.

Then applying the missing word checking on solutions produced by step 7, the match succeeds for the first solution (see the second solution in Table 9). However, it
fails with \{فت\}, leading to discard the last two solutions, yielding the final breakdown of the erroneous word \(قَتَّر\)/qAlo-tw/ (I-told).

- Null prefix, first person singular perfect verb in active voice, and extra suffix Waw.

We conclude from this section that constraint relaxation and edit distance techniques are able to split successfully the erroneous word into three segments: \(prefix + stem + suffix\). In the next section we show how these segments are used in error analysis and feedback generation.

7.2.3 Lexical error analyzer

The lexical error analyzer is the most elaborate of the modules that needs deep Arabic natural language analysis and generation. Heuristics are applied to get rid of irrelevant solutions. Figure 5 presents its architecture. This section explains the components and function of the lexical error analyzer.

Morphological analysis and generation: The final analysis of every word in the learner’s answer is computed by morphological analysis and generation tasks. The morphological analysis is a compound task that takes input from two main NLP tools. The first tool produces a list containing all possible word breakdowns as explained in the previous section. This tool accepts either well-formed or ill-formed Arabic words. The second tool is the online Xerox morphological analyzer, which, like most Arabic morphological analyzers, accepts only well-formed Arabic words, producing all possible analyses of the words having the same root as the correct answer. The decision of whether to accept the stem as a correct analysis of the input word depends on the success of morphologically generating this stem backwards from the root in this analysis using its intended features as specified in the answer associated with the authored question.
The following steps are used to get the final word analysis:

1. Combine the word analysis with the morphological analyzer output
2. Filter the solution list using heuristic rules.
3. Apply morphological generation
4. Compare the generated stem against the analyzed stem
5. Suggest the final word analysis using heuristic rules

**Step 1. Combine the word analysis with the Xerox morphological analysis**

The main purpose of this step is to form rich morphological representations by combining the results of both word analyzer and Xerox morphological analyzer. Xerox morphological analyzer is considered to be one of the most respected tools that we rely upon for performing morphological analysis of well-formed words, which might form a wrong answer. It is obvious that a leaner incorrect answer might include either well-formed or ill-formed erroneous input, and this was the reason that we developed our word analysis tool. In the example that we use for illustration, the morphological analyzer does not produce any solution. Hence, the combined analyses of the ill-formed word /qAlo-tw/ (I-told) is the solution produced by the word analyzer, which is ‘first person singular perfect verb in active voice with extra Waw in the suffix’.

**Step 2. Filter the solution list using heuristic rules**

The solution list could be improved if we consider a number of factors: learner answer, question parameters, and error categories handled by the system. We developed a set of heuristic rules that excludes irrelevant solutions. Two examples of these rules are presented in Table 10. Considering the illustrative example, none of the heuristic rules were applicable such that no filtering took place to rule out any irrelevant solution.

**Step 3. Apply morphological generation**

At this step the morphological generation is done as a reverse process using the analyzed structure. In particular, the Arabic morphological generation tool applies features of every so far analysis to the correct root to generate a well-formed Arabic stem. We followed the same morphological generation approach proposed by Cavalli-Sforza et al. (2000) and Soudi et al. (2001) to develop an Arabic shallow morphological generator. This approach has the advantage of reducing the complexity of Arabic morphological generation by employing discrimination tree representation augmented with transformational rules. The morphological generation approach solves the problem of generating a large number of Arabic verbal variants by decoupling the problem of stem changes (infixation) from that of prefixes and suffixes, which has been proved to significantly reduce the number of rules required. A tree is used to build a hierarchical representation that combines the classification of morphological variants. Each internal node of the tree specifies a piece of the feature structure that is common to that entire subtree. The root of the tree is a special node that simply binds all subtrees together. The leaf nodes

---

23 To clarify, if the analysis was successful and produced the root to /q-w-l/, then its features would have been combined with the features of word analysis.
Table 10. Examples of filtering rules

<table>
<thead>
<tr>
<th>Rule ID</th>
<th>Rule Description</th>
</tr>
</thead>
</table>
| 1       | **Rule:** IF the correct verb is not **defective** (does not have a weak final radical) and the **suffix** begins with a weak character that does not **match** with the last character in the learner word  
**THEN** discard this solution  
**Explanation:** this rule comes from Arabic pronunciation and type of errors categories handled by Arabic ILTS. In Arabic, when a verb is connected to a pronoun that begins with a letter similar to a weak letter, the last letter in the verb should have a diacritic sign compatible with this connected pronoun, e.g. Fatha' 'اَي' if the pronoun begins with /ا/, such that when the learner erroneously uses a short vowel instead of a long vowel the later should be the first letter in the connected pronoun.  
**Example:** If the learner writes the word زِرِين /ta-zuri-yna/ (you-(f, sg) visit) instead of زرو /zuro-ti/ (you-(f, sg) visited), the system will extract fifty different suffixes; some of them begin with /ا/ or /و/ characters. According to **Rule-1,** the system should discard these cases (the weak verb is hollow), these suffixes begin with /ا/ or /و/ but the last character in the learner answer is /ي/). |
| 2       | **Rule:** IF the **affix** in the learner answer **matches** the correct affix but their **FS’s** does not **match**  
**THEN** discard this solution  
**Explanation:** this rule comes from the fact that if the learner uses an affix that matches with the correct answer’s affix then s/he most properly refers to the same affix that have the same orthographic form and meaning.  
**Example:** If the learner writes the word قأل /qAlo-tu/ (I-told) instead of قل /qul-tu/ (I-told), the system will extract four suffixes having the same orthographic form /Te/ but differ in their meaning. The interpretations of these suffixes are: first person singular, second person singular masculine, second person singular feminine, and third person singular feminine. According to **Rule-2,** all these suffixes except the first person singular suffix are discarded from the solution list. |

of the tree correspond to distinct morphological forms in the language. Each node in the tree below the root is built by specifying the parent of the node and the conjunction or disjunction of feature-value pairs (e.g. voice: active, type: hollow) that define the node. A morphological transformation rule is attached to each leaf node. The rule consists of one or more mutually exclusive clauses, each of which has two parts. The first part of a clause is matched against the value of the root feature. The second part includes one or more operators that will be applied in the given order. Operators include addition, deletion, and replacement of infixes. For a specified feature structure, its features are matched against the features defining each subtree (e.g. perfect: verb) following a path until a leaf is reached. The output of applying transformations rule at the leaf is the **transformed stem** from its root and other features. An example of rules showing the stem change for perfect hollow verbs of the pattern فَعِل فُعَالْ /faEala/ is shown in Figure 6.

In Figure 6, consider the perfect hollow verbs in active voice, where the stem keeps the long vowel /ا/ as a middle letter, for third person singular (e.g. قَال ‘he/it said’ and قَالَ ‘she-said’), dual (e.g. قَالَا ‘they-m-dual-said’ and قَالَ ‘they-f-dual-said’), and plural masculine (e.g. قَالُوا ‘they-m-pl-said’). Whereas, in the remaining  

---

24 This is because we use three-way-match method, which might overgenerate, e.g. recognizes the suffix 'ت' with deleted letter 'ت'.

---
Analysis and feedback of erroneous Arabic verbs

Fig. 6. A subtree showing stem change for perfect hollow verbs of pattern /faEala/.

person–number–gender combinations the stem takes a short vowel whose diacritics depend on the underlying root of the verb. The transformation rule attached to the leaf node performs the conversion of the vowel (short or long) on the root. For example, applying the morphological generation step on the illustrative example solution, i.e. ‘first person singular perfect verb active voice with extra Waw in the suffix’, the rule deleteMiddleWeakLetter is triggered. This rule converts the weak middle letter in the root لـ into a short vowel and generates the stem قـ /qul/; see the following FS:

Word: قـ
Root: ـلاـ
Lexical category: verb
verb type: hollow
Pattern: فعل
Tense: perfect
Voice: active
Mood: 
Subject person: 1
Subject num: sg
Subject gender: neutral
Object person: 
Object gender: 
Object num: 


Step 4. Compare the generated stem against the analyzed stem

In this step, a well-focused three-way-match method is used to parse the generated stem against the analyzed stem. The insertion and deletion of letters take into consideration the weak letters and the conversion of letters considers those that have similar pronunciation. The matching process works as follows: Partition the two stem strings: generated stem = xuz and extracted stem = xz, where x is the leading segment, z is the trail segment, and u and v are the error segments. First, the leading segment is selected, which might be empty if the leading characters of generated stem and extracted stem are different. In an exact match between these two stems, this segment should contain the whole word. Second, the trail segment is selected, which might be empty if the last characters of both stems are different. Finally, the error segments are the remaining characters of the two stems. The three-way-match method is applied on these two segments to identify any missing, extra, and converted characters.

Consider current solution of answer (e) in Example 6, the analyzed stem is قل/qAl/ and the generated stem is قل/qul/. Applying this step on these stems yields the leading segment {ق}/q/ and the tail segment {ل}/l/. Hence, the error segment is the middle character {ال}/A/, which is missing for the generated stem قل/qul/. Therefore, the system identifies an extra character {ال}/A/ in قل/qAl/. This result is represented by the structure insert("ال", 2,stem) that indicates that there is extra character(s) in the stem. Table 11 shows the updated error indication.

Step 5. Suggest the final word analysis using heuristic rules

Ambiguity is a major problem in NLP. In our case we are exposed to one-to-many ambiguity where multiple interpretations are observed for ill-formed to well-formed transformations. In the proposed Arabic ILTS, relaxing the linguistic constraints in order to analyze a learner answer generally produces more interpretations than systems designed specifically for handling well-formed input. Consider, for example, the erroneous word عَيَنْتُ / Eiy$ tu/ and the generated word عَيَنْت / Eay-ա$ tu/. This word has two interpretations for error corrections: (1) عَيَنْتُ / Eiy$ tu/ (I-lived), which is related to problems with vowel letters, confusion that led to using of long vowel أ /y/ instead of short vowel أ /i/ “، or (2) عَيَنْت / Eay-ա$ tu/ (I-sustained), which is related to problems with

Table 11. The word analysis of the word قل/qAl-tw/ (I-told) within applying the error analysis

<table>
<thead>
<tr>
<th>Solution Description</th>
<th>Solution Structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prefix: ['', Pref-0, [lexical_category: neutral]]</td>
<td></td>
</tr>
<tr>
<td>Stem string: قال</td>
<td></td>
</tr>
<tr>
<td>Error indication: [insert(&quot;ال&quot;, 2, stem)]</td>
<td></td>
</tr>
</tbody>
</table>
verb pattern, confusion that led to applying the pattern قتل /faE~al/ instead of /faEala/.

In general, NLP approaches usually resolve the ambiguity problem by incorporating disambiguation techniques that suggest a preferred solution. In Arabic ILTS, the preferred method is pedagogical and more closely follows the approach used by the language teacher, who considers the likelihood of an error. The likelihood of an error takes into account the level of instruction and the frequency and/or difficulty of the taught Arabic concepts. Our main concern here is to avoid misleading or incorrect feedback. In Arabic ILTS, we decided to handle three types of ambiguous verb analysis that are related to erroneous learner input:

1. When the Arabic verb conjugation imposed by the passive voice and imperative tense has the same orthographic verb form imposed by the active voice and imperfect tense, respectively. The learning skill level can be used to disambiguate this type. Note that it is doubtful that a beginner student of Arabic writes a passive voice verb instead of an active because the passive voice is a rare construction, even at a more advanced level.

2. When a verb affix has different feature-value pair interpretations. For example, the suffix /Teh/ can be interpreted in four ways: (1) first person singular, (2) second person singular masculine, (3) second person singular feminine, and (4) third person singular feminine perfect verb suffixes.

3. When verb patterns share the same orthographic form but differ in vocalization. For example, verb conjugations from the root following the pattern قتل /faEala/ (for a verb indicating did) and قتل /faE~al/ (for a verb indicating activated) are different.

In the reset of this section we will show how Arabic ILTS deals with these types of ambiguity. In case of the first type of ambiguity, the system selects the word analysis that a student most likely intended. It implements two prioritized conditions to select the most preferred word analysis. The first condition: If the question goal is to test passive voice, then the system selects passive voice analysis, otherwise it selects the active voice analysis. The other condition: If the question goal is to test imperative tense, then the system selects imperative tense analysis, otherwise it selects the perfect or imperfect tense analysis. For example, given the following learner input:

a. /ta-biAE jad~atiy Al>aruz~/ (My-grandmother sells the-rice).

b. /ta-biyE jad~atiy Al>aruz~/ (My-grandmother sells the-rice).

The learner has incorrectly conjugated a hollow verb in the imperfect tense. The error analyzer produces two possible analyses: third person singular feminine imperfect active verb and third person singular feminine imperfect passive verb. Given the question goal that tests conjugation and usage of imperfect verb active voice, the prioritized condition applies that selects the active voice analysis.

In case of the second ambiguity type, the system merges all of the affix analyses in one structure. For example, given the following learner input:

a. محمد خورشيد /muHam~ad tawar~aTt fiy jariyimp qatol/ (Mohammed was-involved in murder crime).
Table 12. Four analyses of the word توارت/tawarẓ-aṬωr/ due to one-to-many ambiguity of the suffix ت/Teh/

<table>
<thead>
<tr>
<th>Solution Description</th>
<th>Solution Structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>1- First person singular perfect verb in active voice</td>
<td>Prefix: (&quot;,Pref-0, [lexical_category: neutral])</td>
</tr>
<tr>
<td>Error indication: []</td>
<td></td>
</tr>
<tr>
<td>2- Second person singular masculine perfect verb in active voice</td>
<td>Prefix: (&quot;,Pref-0, [lexical_category: neutral])</td>
</tr>
<tr>
<td>Error indication: []</td>
<td></td>
</tr>
<tr>
<td>3- Second person singular feminine perfect verb in active voice</td>
<td>Prefix: (&quot;,Pref-0, [lexical_category: neutral])</td>
</tr>
<tr>
<td>Error indication: []</td>
<td></td>
</tr>
<tr>
<td>4- Third person singular feminine perfect verb in active voice</td>
<td>Prefix: (&quot;,Pref-0, [lexical_category: neutral])</td>
</tr>
<tr>
<td>Error indication: []</td>
<td></td>
</tr>
</tbody>
</table>

b. محمد تورت في جريمة قتل /muHam¬ad ta-war¬aTa fiy jarjimap qatol/ (Mohammed was-involved in murder crime).

The learner here has made a subject–verb disagreement between محمد (Mohammed) the inchoative (متيت) of the sentence and its enunciative (خیر). The error analyzer produces four possible analyses, as shown in Table 12. This step will merge the four analyses into one generic solution that combines the structure of affixes, as shown in Table 13. Across the whole analyses, common (identical) feature-value pairs, e.g. subj number: singular, will have an exact instance in the new structure and copied as the same. Whereas fully covered values, e.g. subject gender: masculine, feminine, or neutral, will have a do-not-care value in the new structure indicated by the special symbol ‘$’ i.e. subject gender: $.

In case of the third type of ambiguity, the system merges all of the stem analyses in one structure. For example, given the following learner input:
Table 13. Resolving suffix ambiguity by merging closely related solutions in Table 12

<table>
<thead>
<tr>
<th>Solution Description</th>
<th>Solution Structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>1- singular Perfect verb in active voice</td>
<td>Prefix: (',Pref-0,[lexical_category: neutral])</td>
</tr>
<tr>
<td>Stem string: نَروَدت</td>
<td></td>
</tr>
<tr>
<td>Error indication: []</td>
<td></td>
</tr>
</tbody>
</table>

Table 14. Two analyses of the word /ناقـُـاـلَعَـعَأ/ (moved) due to one-to-many ambiguity of the pattern orthography

<table>
<thead>
<tr>
<th>Solution Description</th>
<th>Solution Structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>1- Third person masculine plural Perfect verb in active voice</td>
<td>Prefix: (',Pref-0,[lexical_category: neutral])</td>
</tr>
<tr>
<td>Stem string: نَقل</td>
<td></td>
</tr>
<tr>
<td>Error indication: []</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Solution Description</th>
<th>Solution Structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>2- Third person masculine plural Perfect verb in active voice</td>
<td>Prefix: (',Pref-0,[lexical_category: neutral])</td>
</tr>
<tr>
<td>Stem string: نَقل</td>
<td></td>
</tr>
<tr>
<td>Error indication: []</td>
<td></td>
</tr>
</tbody>
</table>

a. جَدِي و جَدِيَّة نَقَلْوُا إلى بَيْت جَدِيد / jad-iy wajad-~apiy naq-~aluwA <ilaY bayot jadiyd / (my-grandfather and my-grandmother moved to a new house).

b. جَدِي و جَدِيَّة نَقَلْوُا إلى بَيْت جَدِيد / jad-iy wajad-~apiy {inotaqalA <ilaY bayot jadiyd/ (my-grandfather and my-grandmother moved to a new house).

The learner here has made two errors: (1) subject–verb disagreement between the inchoative (نَقَلْوُا) of the sentence جَدِي و جَدِيَّة and its enunciative (خير) نَقَلْوُا, and (2) incorrect conjugation of a perfect verb that follows the pattern /فَعَل/ /faEala/ instead of the pattern /فَعَل/ /faE~al/. The error analyzer produces four possible analyses as shown in Table 14. This step will merge the two analyses into one generic solution that combines the structure of stems as shown in Table 15. The only difference is the use of do-not-care value with the pattern feature.

Error Detection: This module identifies different error types from the word analysis structure. It relies on production rules to recognize these error types. Figure 7 shows some examples of these rules.

Note that the learner might make multiple errors in his/her input. So this module exhaustively tests all rules to detect all possible error types the learner has made.
Table 15. Resolving suffix ambiguity by merging solutions in Table 14

<table>
<thead>
<tr>
<th>Solution Description</th>
<th>Solution Structure</th>
</tr>
</thead>
</table>

Rule (Verb Tense error):
IF Verb tense of the correct answer does not match tense of the analyzed verb THEN the Detected Error is in tense with parameters [(correct tense, correct voice, correct mood) (analyzed tense, analyzed voice, analyzed mood)].

Rule (Verb Conjugation error):
IF stem is corrected by a character insertion OR (stem is corrected by a character conversion which belongs to either {ؤ،و} or {،ي}) THEN the Detected Error is in verb conjugation with parameters [verb type, analyzed tense, analyzed mood, analyzed voice, type of connected pronouns either consonant or vowel].

Accordingly, it should also update the student model by incrementing the EFC variable for only error knowledge associated with the error made by the learner. This is done by searching the student model database to get the error knowledge index associated with the error made by the learner and then increment the EFC variable associated with this index.

Applying these rules on the word analysis of /qAlo-tw/ (I-told), i.e. answer (e) in Example 6 shown in Table 11, the system detects the following errors:

1. Verb tense error since the tense of the correct verb is imperfect whereas the tense of the analyzed verb is perfect.
2. Long vowel instead of short vowel error since there is an extra suffix character.
3. Verb conjugation error since there is an extra character at position 2 in the stem.

These three errors with their parameters will be sent to the error scheduler module.

Error scheduler: The last step in analyzing learner’s answer is handled by the error scheduler module. The task here is to accommodate multiple errors, instructional feedback messages need to be prioritized by the system and displayed one at a time to the student to avoid multiple error reports. While it is desirable to construct a system capable of detecting and accurately explaining all errors, it does not follow that the system should display each and every error detected.

The proposed Arabic ILTS maintains an error priority queue which determines the order in which instructional feedback messages are displayed to the learner. It ranks instructional feedback with respect to the dependency of errors, e.g. verb tense error has higher priority than verb conjugation error.
Applying this module on the word /qAlo-tw/ (I-told), the error priority queue gives a higher priority to verb tense error over the other two errors (i.e. verb conjugation, long vowel error instead of short vowel error). Finally, this error type (verb tense) as well as its parameters ([(‘imperfect’, ‘active’, ‘indicative’) (‘perfect’, ‘active’, ‘‘)]) is to be sent to tutoring module.

8 Evaluation and results

It is necessary to evaluate the error diagnosis techniques to demonstrate the capability of Arabic ILTS in diagnosing errors made by SLLs of Arabic. The quantitative measures are used to see how far the system successfully diagnoses errors and provides correct error-specific feedback that conforms to the learning skill level. The evaluation methodology relies on the following steps:

1. Acquire test set. The source of this test set directly comes from the text written by real SLLs in their typical learning environment. The test set should contain competence and performance errors produced by learners of the language.

2. Subjectively identify the source and type of errors. An Arabic human specialist analyzes the collected material by recognizing different error types and their source of errors, and annotates the test set with this information. The annotated test set is used as a reference set.

3. Conduct the evaluation experiment. The developer of the system will feed the unannotated test set into the error diagnosis component. A report of the detected and undetected errors will be issued. The recall rate for each error type and the precision ratio of the system will be used to calculate the system’s performance; a high recall rate indicates that a large percentage of errors has been identified, while a high precision ratio confirms that the system detects real errors and does not produce much incorrect errors.

4. Discuss the evaluation results. The evaluation results will be analyzed and the problems where the system failed to identify the exact source of error will be classified.

In the rest of this section we show how we implement this methodology.

8.1 Data collection and error tagging

In the absence of a complete computationally erroneous learner corpus for Arabic that can be used to evaluate the proposed approach and techniques, we have to manually collect the test set from real teaching environment. This corpus should be manually annotated with learner errors information. A real test set that consists of 116 real Arabic sentences is acquired. These sentences are collected from two different sources (i.e. different teaching environments with different Arabic SLLs background): (1) the Arabic Language Institute, at the American University in Cairo (AUC), where Arabic is taught to different learners from different countries, and (2) PhD dissertation that examines Arabic verb errors made by Malay’s SLLs of Arabic (Jassem 2000). The number of words per sentence in the whole test set
Fig. 8. Example of error annotation in XML format.

varies from three to fifteen, with an average of 5.1 words per test sentence. The total number of words in all test sentences is 587, of these 118 words have lexical verb errors. Moreover, thirteen verbs fall under multiple lexical errors; eleven out of these thirteen have two errors whereas only two verbs have three errors.

Creating a large annotated learner corpus as a test dataset is expensive and beyond the scope of this research project. It is much related to Corpus Linguistics that has research methods to study data collection of representative elements agreeing on tag set, manual annotation, inter-annotator agreement, verification, and so on. Although the acquired test set is relatively small, we have shown that it is sufficient to demonstrate 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.

After collecting this test set, the raw text of the test set is judged by Arabic human specialist who manually identified and classified lexical errors. These errors are annotated in the test set; 133 errors were identified and classified into nine error types that belong to one hundred sources of errors. Figure 8 shows an example of annotated reference set used in evaluation. For clarity purposes, the annotation is presented using XML format.

8.2 The experiment

The unannotated test set is fed into the system and the output (i.e. error-specific feedback messages that conform to the learner expertise level) is compared against the manually annotated reference set. The same annotator who performed the test set annotation also compares the reference set with the system’s output. Note that our system provides feedback that conforms to the learning level. So the feedback to the beginner learning level is compared with the annotated source of error in the reference set, while the feedback to the intermediate learning level is compared with the annotated error type. The feedback to the advanced learning level is compared with the annotated class of error.
Table 16. Results of applying the proposed Arabic ILTS on real test set

<table>
<thead>
<tr>
<th>Error type</th>
<th>Freq.</th>
<th>Freq.</th>
<th>%</th>
<th>Freq.</th>
<th>%</th>
<th>Freq.</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Consonant letters</td>
<td>8</td>
<td>8</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2. Vowel letters</td>
<td>24</td>
<td>19</td>
<td>79.17</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>20.83</td>
</tr>
<tr>
<td>3. Verb conjugation</td>
<td>21</td>
<td>14</td>
<td>66.67</td>
<td>1</td>
<td>4.76</td>
<td>6</td>
<td>28.57</td>
</tr>
<tr>
<td>4. Connected pronouns</td>
<td>7</td>
<td>6</td>
<td>85.71</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>14.29</td>
</tr>
<tr>
<td>5. Verb pattern</td>
<td>14</td>
<td>8</td>
<td>57.14</td>
<td>3</td>
<td>21.43</td>
<td>3</td>
<td>21.43</td>
</tr>
<tr>
<td>6. Lexical category</td>
<td>16</td>
<td>14</td>
<td>87.5</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>12.5</td>
</tr>
<tr>
<td>7. Verb tense</td>
<td>17</td>
<td>11</td>
<td>64.71</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>35.29</td>
</tr>
<tr>
<td>8. Subject–verb disagreement</td>
<td>24</td>
<td>17</td>
<td>70.83</td>
<td>5</td>
<td>20.83</td>
<td>2</td>
<td>8.33</td>
</tr>
<tr>
<td>9. Verb mood</td>
<td>2</td>
<td>2</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>133</strong></td>
<td><strong>99</strong></td>
<td><strong>74.44</strong></td>
<td><strong>9</strong></td>
<td><strong>6.77</strong></td>
<td><strong>25</strong></td>
<td><strong>18.79</strong></td>
</tr>
</tbody>
</table>

The system’s performance is measured in terms of recall rate for each error type and the precision ratio of the system; a high recall rate indicates that a large percentage of errors have been identified, while a high precision ratio confirms that the system detects real errors and does not overflag much. Note, however, the following four calculations should be made for each error type handled by the system: (1) A precise ratio of the number of errors in the test set; (2) the number of errors which has been successfully diagnosed by the system (i.e. the source of error is identified), (3) the number and types of errors which the system has successfully detected but failed to identify their exact source (i.e. partially diagnosed errors). For example, the system might detect subject–verb disagreement but fails to identify the source of this error such as disagreement in person, gender, and number, and (4) the number of errors which have been successfully detected but failed to identify their types (i.e. general error indication).

8.3 Analysis of results

Table 16 shows experimental results. The first column describes different error types handled by the system, while the second column presents the frequency of each error type in the test set. The rest of columns present the recall rate of complete diagnosis, partial diagnosis, and general error indication, respectively. Note, however, that in the case of partially diagnosed errors, the system produces the same error-specific feedback message. The reason is that the system was not able to detect the source of error, i.e. only the error type is detected. The feedback message in the case of general error indication is a catch-all error message for all learning levels (there is an error in the word …). The recall rate metric is used to measure the system performance. The recall rate is shown in the last raw. It is calculated by summing up all the diagnosed errors for each of the last three columns and then dividing each
of them by the total number of errors (133 errors) in the test set, e.g. 99/133. The precision ratio is 100 percent (i.e. 99/99), it is calculated by summing the number of correctly diagnosed errors by the system, and then dividing it by the total number of errors diagnosed by the system, including the cases of erroneously diagnosed errors (i.e. overflagging or overdetection cases). Note, however, that our system is designed such that it never overflags, since the first step in the system is to match correct answer against the learner answer such that it does not identify correct input by the learner as an error.

Following can be observed in Table 16. The highest frequent errors made by Arabic SLLs were vowel letters and the subject–verb disagreement with a frequency of 24 for each of them. The accumulated frequency of these two errors together was 48 out of 133, which represents 36.09 percent of total errors. The least frequent error is verb mood, which was found twice. It represents 1.5 percent of the total errors found.

The highest recall rate was for both consonant letters and verb mood that achieved 100 percent. The least recall rate was for verb pattern (57.14 percent). This arose because of the errors ambiguity problem, the system has no direct knowledge of what the student meant to express. For example, when the learner's answer includes the word تعلم instead of تعلمت /ta-Eal~am-tu/ (I-study), it is not clear whether the learner meant تعلمات /Ealim-tu/ (I-knew) that has the pattern فعل /faEil/ or تعلمات /Eal~am-tu/ (I-taught) that has the pattern فعل /faE~al/. In this case, the system was able to successfully detect that the error type is verb pattern, but could not identify the wrong pattern. Moreover, the general error indication also arose from ambiguous situations. For example, the word أكمل can have four possible interpretations: the verb >a-komul/ (I-become complete), the verb >u-komil/ (I-complete), the verb >akomala/ (he/it completed), and the noun >akomal/ (perfect).

9 Conclusions and future work

Arabic language is a highly derivational language that makes it a challenge for SLLs. The conjugation of Arabic verbs is one of the challenging areas in learning Arabic due to their richness in form and meaning. Therefore, SLLs need to get help in their learning process. This is the role of ILTS. The novel Arabic ILT design presented in this paper emulates a language instructor by evaluating and responding to student answers to language exercises. The techniques employed generate error-specific feedback adapted to the student expertise.

When designing the system, we found that it is important to differentiate between learners’ proficiency level, in particular beginner, intermediate, and advanced levels. This is because language learners have different needs at different levels of overall proficiency. Beginner and intermediate learners need to be introduced to linguistic concepts and categories and the rules governing correct forms and usage, in addition to vocabulary. Oral and written tasks tend to have narrow focus, targeting vocabulary and forms introduced lesson by lesson. As the student progresses, tasks are designed to allow increasing levels of creativity. Since the system targets beginner
to intermediate SLLs, the objective test method is used such that the expected learner’s answer is relatively short and well focused. Moreover, the test questions are constrained such that the base word forms expected in a correct answer are always known, and the correct order is specified.

An important process in designing ILTS is the selection of errors handled by the system. Therefore, a study of the most common Arabic lexical errors made by SLLs of Arabic has been conducted. This study indicates that SLLs of Arabic tend to make errors related to lexical category selection, pattern selection, tense selection, mood selection, subject–verb agreement, verb conjugation, connected pronouns, consonant letter, and vowel letter. The result of this study has enabled us to design and implement the proposed system that successfully analyzes learner’s answer and provides an error-specific feedback.

The rule-based approach employed here gives some freedom to language learners in the way they phrase their answers. They can produce few words rather than selecting them from multiple choices. Moreover, the rule-based approach enables the exercise author to enter only one possible correct answer, thus saving much time compared to the traditional pattern matching answer-coding approach. The rule-based approach has the advantage of providing detailed analysis of the learner’s answer using linguistic knowledge. This detailed analysis is used to construct an informative feedback that indicates the location of errors and their types. Furthermore, the constraint relaxation and edit distance techniques that have been successfully used in ILTS to diagnose learner’s answer are also used here. These techniques need some context-sensitive information, such as list of affixes and patterns. The finite-state automata approach has been successfully used to represent the required information.

A further pedagogical challenge for ILTS with respect to meaningful feedback is multiple errors made by students in one sentence. While it is desirable to construct ILTS capable of detecting and accurately explaining all errors, it does not follow that the system should display each and every error detected. In the absence of an error filtering mechanism, the sheer amount of feedback would overwhelm students. However, a language instructor typically skips irrelevant errors and discusses the remaining ones one at a time. In the proposed system, therefore, a feedback message displayed to the learner considers the following criteria: (1) It needs to be accurate in order to be of any use by the student, (2) displaying more than one error message at a time is not very useful because at some point they probably will not be read, and (3) explanations for a particular error should also be kept short. According to these criteria, the system maintains an error priority queue that determines the order in which instructional feedback messages are displayed to the learner. It ranks instructional feedback with respect to the dependency of errors: lexical category selection, pattern selection, tense selection, subject–verb agreement, mood selection, connected pronouns, verb conjugation, and/or consonant and vowel letters. Consequently, the proposed system faces the challenge of multiple errors by taking language teaching pedagogy into account.

A further challenge for ILTS is providing individualized tutoring by responding to the specific needs of student, guiding beginner learners, challenging advanced
learners, and monitoring the progress of each student as well as establishing a training plan. In the proposed system, this is done through the student model that records information about the current learning skill level of each student. The use of student model allows the system to provide feedback messages that conforms to the learner’s expertise for each class of errors. The learning level is not absolute for all student model knowledge as it is calculated for each individual error which enables the system to focus on providing guidance that helps the learner in understanding his/her problems and improves his/her learning skill level.

Much remains to be done in the field of lexical checking in the context of ILTS. We are still far from tools that cover all error categories committed by language learners in a robust, reliable, and efficient manner. Thus, there are many paths open for further research. A few are cited below.

The proposed system should be put into a real learning environment and used by real learners. Moreover, the task of diagnosis of free Arabic text enhances the teaching process as a whole, since learners can now produce paragraphs freely and be guided by themselves to recognize by the erroneous or inappropriate functions of their misused expressions. The work in this paper depends on the objective test method, so the system is needed to be expanded to diagnose free Arabic text and provide error-specific feedback messages.

The ambiguity problem is partially solved using the instruction learner level and difficulty of Arabic concepts, but it needs to be completely solved using techniques that more follow the approach of a language teacher who considers the likelihood of an error. The likelihood of an error takes into account the level of instruction and the frequency and/or difficulty of Arabic concepts. The concern here is avoiding misleading or incorrect feedback. Using the same techniques as followed by other NLP applications to disambiguate multiple reading may produce incorrect analysis and mislead the learner.

The proposed system limits to some extent the handling of multiple errors on different error classes, so it is needed to be expanded to handle different lexical error classes (word formation, word choice, and errors at the interface between lexical and grammar levels).

The proposed system uses a student model that follows the perturbation error model to represent this knowledge. In this model, one or more misconceptions were assumed to exist for each concept in an introductory course of Arabic weak verbs. By this way, the student model knowledge is represented by a union of a subset of the domain knowledge (i.e. Arabic concepts of weak verbs) and another subset of the misconception set. This representation needs to be enhanced by the use of analytic methods that use machine learning techniques to represent student misconceptions without requiring listing a set of pre-defined errors or errors.

References


Shaalan, K. 2010, June. Rule-based Approach in Arabic Natural Language Processing, Special Issue on Advances in Arabic Language Processing, the International Journal on Information and Communication Technologies (IJICT), Serial Publications, New Delhi, India, 3(3):11–19.


