Towards Automatic Spell Checking for Arabic

Khaled Shaalan
Computer Science Dept., Faculty of Computers & Information, Cairo Univ.
5 Tharwat St., Orman, Giza, Egypt
shaalan@mail.claes.sci.eg

Amin Allam
Computer Science Dept., Faculty of Computers & Information, Cairo Univ.
5 Tharwat St., Orman, Giza, Egypt
aminallam@yahoo.com

AbdAllah Gomah
Computer Science Dept., Faculty of Computers & Information, Cairo Univ.
5 Tharwat St., Orman, Giza, Egypt.
abdallah_1982@hotmail.com

Abstract

Arabic's rich morphology (word construction) and complex orthography (writing system) present unique challenges for automatic spell checking. An Arabic checker attempts to find a dictionary word that might be the correct spelling of the misspelled or misrecognized word. In this paper, we report our attempt in developing an Arabic spelling checker program for solving this problem. Our approach is heuristic and involves developing an Arabic morphological analyzer, techniques of spelling checking and spelling correction, and efficient methods of lexicon operations. The developed Arabic spell checker is able to recognize common spelling errors for standard Arabic and Egyptian dialects.

1. Introduction

Arabic is a highly inflected natural language that has enormous numbers of possible words (Othman et al., 2003). An Arabic spell checker is a program that tries to check spelling automatically. This device makes it much easier to proofread your documents and catch all errors. Without it you should proofread and correct your documents in the old fashioned way—read them yourself. The use of word processors and text editors leads to a whole class of writing errors (Hahne, 1999). Thus, many popular word processing software accompany spelling checkers. The role of the spelling checker— whether integrated or standalone— is to analyze the word and try to catch these errors.

Arabic spell checking is an active area of research since results are not satisfactory. This work addresses the challenges of creating a general purpose spelling checker for Arabic. It recognizes common Arabic spelling errors and offers suggestions for error correction. This program is useful for anyone who writes: students, business people, and professional and casual writers. Arabic spelling checker is targeted to be part of any Arabic text processing programs such as word processors, web browsers, among others. The tool has been successfully implemented using SICStus Prolog on IBM PC.

The rest of this paper is structured as follows. In Section 2, we give a brief background about the relevant aspects of the Arabic language. In Section 3, we introduce our analysis of the common spelling errors that are used for detecting the misspelled Arabic word. In Section 4, we describe our proposed method for spelling correction. In Section 5, we present efficient methods of morphology and lexicon operations. In Section 6, we give some concluding remarks.

2. Aspects of Arabic Language

Arabic is strongly structured and highly derivational (Kiraz, 2001). Arabic morphology and syntax provide the ability to add a large number of affixes to each word which makes combinatorial increment of possible words (Rafea et al., 1993). In Arabic, there exist some issues, which need to be taken in considerations when designing a spell checker: computational morphology, weak and consonant characters, and morphographemic rules.

2.1 Computational Morphology

Computational Morphology concerns how to derive a new word from an existing one by adding an affix to the original word (Ramsay et al., 2001). The affix may be prefix, infix or suffix. In Arabic there are two types of morphology. The process is explained below with regard to the infix case:

1. Inflectional Infix Morphology: in which the word category doesn't change.
   - وعد (Base) + 'ا' (Infix) → واعد (The new word is also a verb)
   - باع (Base) + 'ا' (Infix) → بايع (The new word is also a verb)
2. Derivational Infix Morphology: in which the word category changes.
   - عمل (Base) + 'ا' (Infix) → عامِل(The word category changed to a Noun)
   - قال (Base) + 'ا' (Infix) → قائل (The word class category to a Noun)

2.2 Weak and Consonant Characters

In Arabic, the weak characters ('ا', 'و', 'ي') and the Hamza character ('ء', 'أ', 'إ', 'ئ', 'ـئـ', 'ؤ') change according to the diacritic sign of the surrounding characters (Buckwalter, 1992). The character in an inflected word is either consonant characters, such as (ضرب – يضرب – يلعب) or weak characters and can be changed such as (قـال – نقول – نسير). The following are the possible changes:

1- The weak character may be deleted from the word, e.g. (قال → قُل), (وقى → يقي), and (قضى → اقض)
2- The weak character may be replaced by another weak character or by Hamza character, e.g. (دعا → يدعو), (قال → قائل), (بات → يبيت), and (آمن → إيمان)

2.3. Morphographemics rules

Some spelling changes are automatically made when when we apply morphology rules such as adding a suffix to a word:

'ات' (suffix) + ضرب (Base) → اضترب → اضطرب ('ت' was replaced by 'ط').
'ات' (suffix) + زهر (Base) → ازتهر → ازدهر ('ت' was replaced by 'د').

3. Analysis of Common Arabic Spell errors

In order to investigate the possibility to develop a computational Arabic spelling checker, we analyzed and classified the common spelling errors that would occur when formulating an inflected Arabic word. In the following, we summarize five sources of spelling errors.

A. Reading Errors: This kind of spelling errors would result when the user types in from a written source, possibly handwritten, such that she/he misrecognizes a character and replaces it with another one that looks like it. From the reading viewpoint, similar characters are grouped into the following categories:
{ ا, أ, إ, آ }, {ب, ت, ث, ـبـ, ـنـ, ـيـ }, { ج, ح, خ }, { د, ذ }, { ر, ز }, {س, ش }, { ص, ض }, { ط, ظ }, {ع, غ }, {ف, ق }, { ه, ة }, {و, ؤ }, and { ي, ى}. The following are examples of this type of spelling errors:

Correct word	Error	Possible Reason
إقامة	اقامة	The character 'إ' is not written on most keyboards.
تساءل	نساءل	The second dot of the character 'ت' is not clear.
جاء	حاء	A dot is missed.
درب	ذرب	A dot is added over the character 'د' because of the pen ink.
رأى	زأى	A dot is added over the character 'ر' because of the pen ink.
سميع	شميع	Dots are taken from the above line.
صدقة	ضدقة	A dot may be missed from the keyboard due to heavy use.
ظافر	طافر	A dot is missed as it is close to the next character
عامل	غامل	A dot may be missed from the keyboard due to heavy use.
قرب	فرب	The two dots are very close to each other such that they appear as a single dot.
هذه	هذة	Dots are taken from the above line.
مورد	مؤرد	A dash over 'و' is considered as a Hamza.
زراعي	زراعى	The final 'ي' is written without dots

B. Hearing Errors: This type of spelling errors would results when the human writer is being dictated; the user may recognize a character as another one. From the hearing viewpoint, similar characters are grouped into the following categories: { ا, ى }, { أ, ق }, { ت, ط, د, ض }, { س, ث, ش }, { ج, ق }, { ذ, ز, ظ }, { ر, ل, ي }, { ك, ق }, and { ة, ه }. The following are examples of this type of spelling errors:

Correct word	Error	Possible Reason
وعى	وعا	The user heard the 'ى' as 'ا'.
قديمة	أديمة	The dictator uses slang Arabic where 'ق' is pronounced as 'أ'.
وعدتهم	وعتهم	The user heard the two characters 'د', 'ت' as one character 'ت'.
ساعد	شاعد	The speaker is very old.
قصر	جصر	The dictator uses upper Egypt dialects where 'ق' is pronounced as 'ج'.
ذبح	زبح	The dictator didn't get his tongue little out when pronouncing the character 'ذ'.
راح	ياح	The dictator has some health problems where 'ر' is pronounced 'ي'.
القدم	الكدم	The dictator does not pronounce 'ق' correctly.
العملة	العمله	The character 'ة' was at the end of the speech and is pronounced as 'ه'.

C. Touch-Typing Errors: This kind of spelling errors would result from a non-experienced human typist due to switching a character with another adjacent one when her/his finger takes wrong position on the keyboard. There are two types of wrong positioning:

1. Shift Right: the right hand is shifted one key to the right, i.e. the right hand is shifted from the original position 'ك', 'م', 'ن', 'ت', to the position 'ط', 'ك', 'م', 'ن'.
2. Shift Left: the right hand is shifted one key to the left, i.e. the right hand is shifted from the original position 'ك', 'م', 'ن', 'ت', to the position 'م', 'ن', 'ت', 'ا'.

The following are examples of this type of spelling errors:

Correct word	Error	Possible Reason
كن	طم	The right hand shifted to the right one key.
من	نت	The right hand shifted to the left one key.

D. Morphological Errors: This kind of spelling errors would result from a nonnative speaker of Arabic or a non well-educated human writer because she/he is not aware of the Arabic morphology. The following are examples of this type of spelling errors:

Correct word	Error	Possible Reason
سألوا	سألو	Plural masculine of the past form of the verb "سأل"
جد	اوجد	Imperative form of the verb"وجد"
ايجع	جع	Imperative form of the verb"وجع"
دعا	دعى	Past form of the verb"دعو"
دعوا	دعيوا	Plural masculine of the past form of the verb"دعو"
اهتدوا	اهتديوا	Plural masculine of the past form of the verb "اهتدى"
يبنون	يبنيون	Present form of the verb "بني"
كفيناكموهم	كفيناكمهم	Past form of the transitive verb "كفي"
يتبارون	يتباريون	Plural masculine of the present form of the verb "تبارى"
اعل	اعلو	Single masculine of the mperative form of the verb "علو"
قاضون	قاضيون	Plural masculine of the noun "قاضي"

E. Editing Errors: This kind of spelling errors would result from typing mistakes due to edit operations such as insertions, deletions, and substitutions. The following are examples of this type of spelling errors:

Correct word	Error	Possible Reason
علم	علمم	The user pressed the character 'م' twice.
استقام	استام	The user forgot to write the character 'ق'.
سمع	سمعغ	The user pressed the characters 'ع', 'غ' with one press.
قام الرجل	قامالرجل	The user forgot to type a space between the two words.
اجتماع	اجمتاع	The user pressed the character 'م' before the character 'ت'.

4 The Proposed Spelling Correction Method

The first step in spelling correction is the detection of an error. There are two possibilities:

1. The misspelled word is an isolated word (Non-word), e.g. 'محد' for 'محمد'
2. The misspelled word is a valid word, e.g. 'مال' in place of 'نال.'

We have limited the detection of spelling errors to isolated words. Once the word is chosen for spelling correction, we perform a series of heuristic steps to find a replacement candidate for it:

1) **Add missing character:** The human writer may have missed a character. The tool tries to add a missing character in every possible position. If the modified word matches a word in the lexicon, it is added to the list of candidates. For example, the candidates of the misspelled word "معض" are "معوض", "معرض", and "معضد".
2) **Replace incorrect character:** The human writer may have typed in or heard a wrong character. The tool tries to replace every character with one of its neighbors according to the table below. A neighbor's character is either an adjacent character in the keyboard, a similar character that either looks like it, or have the same pronunciation. If the modified word matches a word in the lexicon, it is added to the list of candidates. For example, the candidates of the misspelled word "معض" are "معد", "كعض", "نعض"

Character	Neighbors					
ء	ؤ	ئـ	ا	أ	إ	آ
ئـ	ء	ؤ	ا	أ	إ	آ
إ	لإ	ئـ	ا	أ	آ	
أ	لأ	ا	أ	إ	آ	
ا	ت	ل	أ	إ	آ	ى
ؤ	ء	ر	ئـ	إ	أ	آ
آ	لآ	أ	ا	إ	أ	
ب	ي	ل	ت	ث	ن	
ت	ا	ن	ث	ي	ب	
ث	ص	ق	ت	ث	ن	ي
ج	د	ح	خ			
ح	خ	ج				
خ	ح	ه	ج			
د	ج	ذ	ض	ت	ة	
ذ	د	ز				
ر	ؤ	لا	ز	ي		
ز	و	ظ	ر	ذ		
س	ش	ي	ث			
ش	س					
ص	ض					
ض	ص	د				
ط	ك	ظ	ت			

Character	Neighbors					
ظ	ز	ط				
ع	ه	غ				
غ	ف	ع				
ف	غ	ق				
ق	ف	ث	ك			
ك	م	ط	ق			
ل	ب	ا				
م	ك	ن				
ن	م	ت	ب	ث	ي	
ه	ع	خ	ة	ت		
و	ة	ز	ؤ	ا	ى	ي
ى	لا	ة	ا	ي	و	
ي	س	ب	ت	ث	ن	ى
ة	ى	و	ه			

3) **Remove excessive character:** The human writer may have typed in an extra character. The tool tries to delete a character from every possible position. If the modified word matches a word in the lexicon, it is added to the list of candidates. For example, the candidates of the misspelled word "معض" are "عض", "مع".
4) **Add a space to split words:** The human writer may forget to leave a space between two words. The tool tries to add the space in every possible position. If the modified word matches a word in the lexicon, it is added to the list of candidates. For example, the candidates of the misspelled word "معض" are "عض", "مع".

5. Morphological Analysis and the Lexicon

As Arabic is a highly inflected language, we need to provide methods that are capable to accelerate the lexicon lookup and the morphological analysis process at runtime. In this section, we will present efficient methods of storing and looking up Arabic stem.

In our approach, we distinguish between two types of lexicons: *Base Lexicon* and *Stem Lexicon*. The *Base Lexicon* includes primitive word forms and is used to build the *Stem Lexicon*. The *Stem Lexicon* includes partially inflected Arabic words. This Lexicon provides efficiency in storing and looking up entries during the spell checking process because the morphological analysis is simplified.

The *Base Lexicon* includes Arabic roots such as (ق-و-ل), nouns that cannot be generated from their roots by regular morphological rules such as (قلم- كتاب-شجرة), and particles— each with a different set of features. The *Base Lexicon* entry is represented as a Prolog term. The following examples show the representation of the words 'كتاب' and 'س ءل':

['كتاب', word_list([word('كتاب','اسم','مفرد','مذكر', 'غير عاقل', 'تكرة', ' يمكن تأنيثه بالتاء', ' يمكن زيادة ألف تنوين' ,'جامد' ,' يمكن إضافته' ,'لا يمكن أن يصف ' ,'لا يمكن جمعه سالما' ,' يمكن تثنيته' ,'يرفع أو يجر بدون الإضافة' ,'يرفع أو يجر مع الإضافة' ,'إعراب')' يمكن النسب له' ,Affixes, ('يمكن أن تأتى منفردة' ,'لا ينصب بدون الإضافة' ,'ينصب مع الإضافة')])].

['س ءل', word_list([word('س ءل','مادة','منصرف', 'معرب', 'لا يمكن وزنه على أفعل فعلاء ' ,['س', 'ء', 'ل'] ,'تام' ,'ثلاثي' ,'فتح' ,'لا يمكن التعجب منه ' ,'متعدى لمفعول' ,'لازم')'التعدى' ,' يمكن بناؤه للمجهول' ,'فعل مجرد' ,(_ ,'اسم مفعول'),(_ ,'اسم فاعل')] ,' يمكن بناؤه للمجهول' ,'فعل مزيد - ا')),((' ليس متعدى لثلاث مفاعيل' ,'ليس متعدى لمفعولين' ,'متعدى لمفعول' ,'ليس لازم')'التعدى' ,'ليس متعدى لمفعولين' ,'ليس متعدى لمفعول' ,'لازم')'التعدى' ,' يمكن بناؤه للمجهول''فعل مزيد - تا') ,((' ليس متعدى لثلاث مفاعيل')' لايمكن أن يأتى منفردة' ,Affixes,)])].

The *Stem Lexicon* includes partially inflected Arabic words. These entries are generated by means of partially morphological generation procedure. The *Stem Lexicon* entry is represented as a Prolog term. When we apply the partially morphological generation procedure to the word 'كتاب', it will have the same representation as the one in the base lexicon. However, when we apply this procedure to the word 'س ءل', partially inflected word forms will be generated such as the word 'مسئول':

['مسئول', word_list([word('مسئول','اسم', 'مفرد','مذكر ', 'محايد', 'تكرة' ,' يمكن تأنيثه بالتاء' ,' يمكن زيادة ألف تنوين' ,'مشتق' ,' يمكن إضافته' ,' يمكن أن يصف ' ,' يمكن جمعه سالما' ,' يمكن تثنيته' ,'يرفع أو يجر بدون الإضافة' ,'يرفع أو يجر مع الإضافة' ,'إعراب')' يمكن النسب له' ,Affixes, ('يمكن أن تأتى منفردة' ,'لا ينصب بدون الإضافة' ,'ينصب مع الإضافة')])].

For efficiency reasons, we represent the lexicon as a letter tree (Covington, 1994). The following shows a skeletal letter tree for the words (سلام - تمرة - تمادى):

```
ltree([[ت, [م, [ا, [د, [ى,  props([word(p1:تمادى..p2:'اسم'...),
                                   word(p1:تمادى..p2:'فعل'...)]) ]]],
                        [ر,[ة,      props([word(p1:تمرة..p2:'اسم'...)])]]],
                  [س, [ل, [ا, [م,     props([word(p1:سلام..p2:'اسم'...)]) ]]]],
                    ....
                ]).
```

A letter tree is represented in Prolog as list of branches. A branch is a list. The first element of the list is a letter. Each succeeding element is either another branch, or the feature structure of the word. The elements are in a specific order: the lexical entry (if any) comes first, and branches are in alphabetical order by their first characters. The words are stored in alphabetical order for efficiency purposes.

The spell checker module will use a simple morphological analyzer to ensure that the input word is a well-formed inflected Arabic word. For example, the morphological analysis will broke down the inflected word 'المسئولين' into the prefix 'ال', the suffix 'ين', and the stem 'مسئول'. This stem is correct because it has an entry in the stem lexicon.

6. Conclusion

In this paper, we report our attempt to develop Arabic spelling checker. This tool is capable of recognizing and suggesting correction of ill-formed input for common spelling errors. It is composed basically of Arabic morphological analyzer, lexicon, spelling checker, and spelling corrector. We have implemented the Arabic spelling checker tool using SICStus Prolog on IBM PC. The interface is built using Microsoft Visual Basic. This tool is very useful for automating the proofreading of the human typed Arabic text. It can be integrated with other text processing software, such as word processors.

References

1. Buckwalter T. (1992). Orthographic Variation in Arabic and its Relevance to Automatic Spell-Checking, *In the Proceedings of the 3rd International Conference and Exhibition on Multi-lingual Computing (Arabic and Roman Script),* University of Durham, UK
2. Covington, M. (1994). *Natural Language Processing for Prolog Programmers*, Prentice Hall.
3. Hahne H. (1999). Writing Tools, in Using a Computer in Biblical and Theological Studie*s*, *Tyndale Seminary*, Toronto. Available at http://www.balboa-software.com/hahne/harry.html
4. Kiraz G. (2001). *Computational Nonlinear Morphology: with Emphasis on Semitic Languages*, Cambridge University Press.
5. Othman E., Shaalan K., and Rafea A. (2003). A Chart Parser for Analyzing Modern Standard Arabic Sentence, *In proceedings of the MT Summit IX Workshop on Machine Translation for Semitic Languages: Issues and Approaches*, New Orleans, Louisiana, U.S.A.
6. Ramsay A., Mansur H. (2001), Arabic Morphology: A Categorical Approach, *In the proceeding of* Arabic *NLP Workshop at ACL/EACL.*
7. Rafea A., Shaalan K. (1993). Lexical Analysis of Inflected Arabic words using Exhaustive Search of an Augmented Transition Network, *Software Practice and Experience*, 23(6):567-588, John Wiley & sons, U.K., June.