Contribution to the Achievement of a Spellchecker for Arabic

Khaireddine Bacha, Mounir Zrigui

University of Tunis, High School of Sciences and Techniques of Tunis,
Laboratoire LaTICE, Tunisia
{khairi.bacha@gmail.com, mounir.zrigui@fsm.rnu.tn}

Abstract. The objective of this work is to perform a spell check tool that analyzes the text entered in search for possible misspellings. This tool will suggest possible corrections for each misspelled word in the text. This work will require the presence of a reference dictionary of words in the arabic language. These objective Were Accomplished with resources, effective methods, and approaches. First experimental results on real data are encouraging and provide evidence of the validity of the design choices. They also help to highlight the difficulty of the task, and suggest possible developments.

Keywords: Spellchecker, Arabic, dictionary, error detection.

1 Introduction

Natural Language Processing (NLP) is a discipline that closely associates linguists and informati-cians. It is based on language, formalisms (representation of information and knowledge in formats interpretable by machines), and computer science. This is the set of methods and programs that allows computer processing of language data, but when this treatment takes into account the specific-ficités of human language. There are language data processing (writes files, backups or other) that are not part of natural language processing [1]. Indeed, it is integrated with infor-matic tools used daily by millions of people worldwide. A spellchecker detects, in a given input text, the words that are incorrect. A spell checker detects the same time spelling errors and look, for the correct word most likely [2].

Construction of automatic text correction systems is one of the oldest applications of Natural Language processing techniques, since, according to Mitton [3], the first systems of automatic detection appeared in the late 50s. A spell checker performs two essential functions, one after another: first detecting, then correcting spelling errors. Methods for detection and correction work in three ways: First error detection orthographically consisting of foreign words in the language. Then the whole word of correction is to correct the word previously detected in the single recital regardless of the words that surround it. Finally, the detection and contextual error correction where each word is considered taking into account the context; Which corrects spelling mistakes even when it consists of words found in the language but are misplaced [4].

It is in this context that our research lies. For this, we will implement an ortho graphic equalizer that analyzes the text entered to find any spelling errors. This tool will suggest possible corrections for each misspelled word in the text. This work will require the use of reference diction, nary of Arabic words [5].

2 State of the Art in Spelling Correction

Spell checking is to find the word (s) nearest incorrect words in a text of a language, based on similarity and distance inter words. Several researchers have studied the problem ; and through their efforts various techniques and many algorithms have been developed [6]. The first studies were devoted to determining the different types of elementary spelling errors, called publishing operations including: insertion, deletion, permutation, and substitution [7].

The main techniques used for the identification of erroneous words in a text are either based on the analysis of n-grams, or about searching a [8] dictionary. An algorithm for the detection based on a dictionary is given by Peterson [9]. Another modeling proposed by Pollock and Zamora [10] is carried out by comparing the alpha-codes with the erroneous word. It is to associate each word in the dictionary with its alpha -code, hence the need to have two dictionaries: one for the words and another for their alpha -codes. They propose to correct those words containing only one error, which form 90-95% of the errors. A particular problem concerns the errors that result in words that exist in the lexicon. According to Mitton [3], this applies to 16% of spelling errors. Oflazer proposed a new approach for the correction of a wrong word that consits in browsing the dictionary controller by calculating for each transition the cut -off edit distance without exceeding the threshold defined in [11] algorithm.

Despite the availability of a set of methods for spell checking, we find out that we do not yet have robust correction software that can handle appropriately all mistakes in the written text, seeing the bad scheduling solutions suggested when correcting.

2.1 Systems for Arabic

The major problem is that the Arabic language is very rich in morphology, and it has many exceptions. Moreover the lack of vowels and words of agglutination make treatment a more difficult task. The most successful achievements for the Arabic language are:

- **BenOthmaneZribi and Zribi (1999)** evoke the special problems to correct Arabic. Words must sometimes be voweled. Moreover, Arabic is an agglutinative language which uses axes and enclitics (pronouns) and proclitics (adverbs, prepositions and conjunctions). In addition, this language contains many lexemes that are very similar to each other. Candidate proposals for the correction of a word can be very numerous. The corrector is accompanied by a morphological analyzer, which cuts shape proclitic, radical and enclitic [12].
- **ShaalanSpell-Checker (2003)**: a correction can detect and correct common spelling mistakes un Arabic based on the technique of N-gram [13].

- **Haddad-Yaseen Spell Checker (2007)**: It is a hybrid model for spell checking and correction of Arabic words, based on the recognition of semi-isolated words [14].
- **Zerrouki-BallaSpell-Checker (2009)** developed a spell checker for the Arabic language based on N-gram [15].
- **Hasan Muaidi and Rashal al Tarawneh Spell-Checker (2012):** It is a simple and flexible spell checker for the Arabic language based on N-gram scores technique (matrix). The recognition rate of the proposed spellchecker reached 98.99% [16].
- **Gueddah, Yousfi and Belkasmi (2012)** proposed a typical and effective variant of edit distance by integrating the frequency matrices editing errors in the Levenshtein algorithm in order to perfect the correction and scheduling error suggestions committed in the seizure of documents in Arabic [17].

2.2 Error Correction Techniques

Situations where we may use detection or automatic correction of spelling errors are very diverse: isolated words or context, common nouns, proper names, labels ... performance errors are not random but systematic such as the inclusion or omission of letters. They note that a list of proposed corrections must be as short as possible and the correct word must appear in such a high order as possible. The spelling correction process: it is looking for a word in a dictionary, and if the word is not there, the search for the most likely words to represent the correct spelling of the word [18]. Error usually single words of correction techniques can be divided into subcategories:

- **Distance of Levenshtein:** This is a mathematical distance measure that gives a similarity between two character strings. It is equal to the minimum number of characters you need to remove, insert or replace to move from one channel to another. It was proposed by Levenshtein in 1966. It measures the similarity between words by computing an edit distance. The edit distance is defined as the minimum number of elementary edit operations needed to transform a wrong word to a dictionary word. Thus, to correct a misspelled word, it retains a set of solutions that requires the least possible editions of operations. It is also known under the names of edit distance or time dynamic deformation, including pattern recognition, especially in voice recognition [19].
  Called Levenshtein distance between two words M and P the minimum cost to go from M to P by performing the following basic operations:
  - Substitution of M character in a character P;
  - Added in M P a character;
  - Delete character Mr.
- **Correction rules:** Yannakoudakis and Fawthrop offer two spellers by rules. One is based on research of similar words in a dictionary, following some error rules; the second searches in the dictionary words that differ by one or two characters of the unknown channel and checks whether an error rule can be applied. [20] Emirkanian Bouchard and uses heuristics to correct spelling errors. The radical words is looked in a dictionary, which also contains information on the correct suffix and suffix frequent errors. A Dictionary of suffixes allows you to find suffixes invalid language [21].

- **The technique of n-grams:** N-grams are n groups of letters constituting a word substring. The most common are bigrams, consisting of two letters and three letters of the trigrams. In general, the techniques of n-grams examine each of the n-grams constituting the input string and looking its presence or its frequency of occurrence in a precompiled table containing statistics on the most frequent n-grams.
  For de Heer, the trigrams are the smallest units which, combined, are significant to the meaning of language. [22] Angell and his colleagues present a method based on common trigrams between the unknown word and dictionary words. The candidate words are found through a dictionary of trigrams which lists all words that contain the same trigram [23].
- **Probabilistic Technique:** The N-gram based technology naturally led in the probabilistic technique both text recognition and spelling correction paradigms. It requires a very large corpus of text in order to establish the table of n-grams. Research techniques in the dictionary only prospect if the input string appears or not in the list of valid words. If the string is missing from the dictionary ; then, it is called erroneous. The dictionary access time becomes prohibitive when the size of the latter exceeds a few thousand words. This problem was addressed in three distinct views through effective search algorithms [24]; via partitioning and organization of dictionaries [25], or via the techniques of editing distance [26] and morphological processing. The most exploited technique to gain access to dictionaries time is the technique of hash. Most existing spellers are semi automatic, assist the user by offering a set of candidates close to the erroneous word [27].

3 Spellchecker implementation

We will see in this part of resource development, implementation and evaluation of our spelling system. We first describe the principle and the overall architecture of our correction. Then we present the different techniques and spell checking tools. We will finish with an evaluation of our system.

3.1 Principle

Our spellcheck prototype must be able to detect misspelled words in a text input and suggest corrections. It addresses a set of operations for each misspelled word: Error detection is often done by considering one by one the words of the text to correct in isolation. Every word of the text is compared with dictionary words. Any word not found in the dictionary is considered wrong.

AL-Mohit Dictionary is a multifunctional electronic dictionary for Arabic. It studied the macrostructure and microstructure of electronic dictionaries of the Arabic language [5]. IT is very rich in grammatical information and meaning and definitions of words. This dictionary can be seen as a structure composed of linguistic objects. Among these objects we can find: the headword, pronunciation, grammatical categories that can have this headword (فعل, اسم, مصدر, صفة ...) definitions, translations, examples. This dictionary, implements the form of a relational database, contains in its first version fifteen entities or files left in two main branches: a verbal branch and nomina l branch. When an error is detected, correction selects a series of words likely to be the correct version of the string to correct. These words are selected using various techniques.

Generation of possible corrections can be done using the concept of Levenshtein distance, consulting a predefined rule base and adding spaces in the middle of each misspelled word. The authorization of the correction to candidates chains considers the measure used in the selection step, as well as statistical measures (such as the frequency of occurrence of words, or the word most frequently chosen during preliminary meetings with the same error). Finally, interactive step allows the user to supervise the correction. It can adopt one of two attitudes: First, correct the erroneous word by selecting one of the candidates proposed by the checker. Then, modify the wrong word.

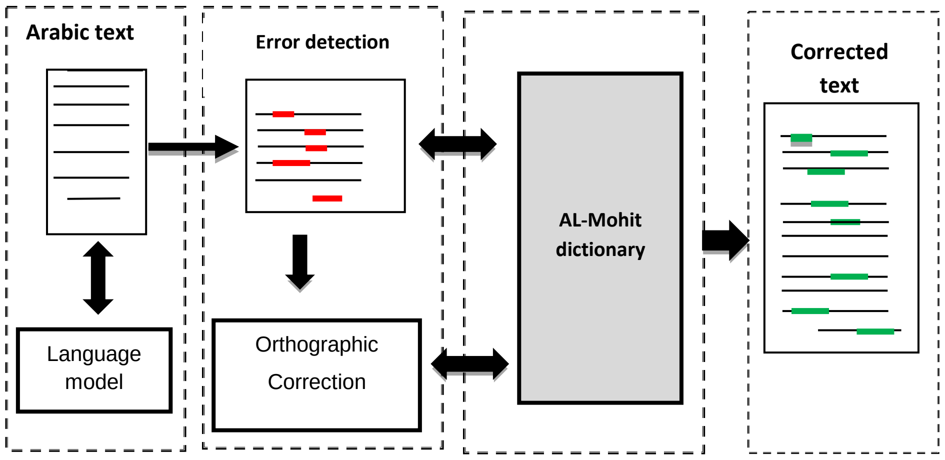


Fig. 1. General architecture of the prototype: spellchecker

3.2 Global architecture

In this section, we briefly present the resources that have been made to develop and test our first spellchecking prototype: It seems impossible to have a large corpus, which represents all possible co-occurrences and all the vocabulary of a language. To solve this problem, we calculate probabilities based on a history of reduced size model called the n-gram. An n-gram is a subsequence of n items constructed from a given sequence. From a given sequence of words it is possible to obtain the likelihood of the occurrence of the next word function. From a training corpus, it is easy to construct a probability distribution for the next word with a history of size n [28]. This modeling is actually an order Markov model n where only the last n observations are used to predict the next word. And a bi-gram is an order Markov model 2 [29].

Language modeling means finding the most likely word knowing those preceding it. This task is performed during the training phase of the corpus of the target language. There are two tool boxes commonly used for building language models, the SRILM tool [30] and the tool CMU-Cambridge [31].These two jewels have open access; and similar features.

3.3 Methods of Error Detection

A natural method to simplify the design of a dictionary is to consider a corpus of documents, assuming that all documents of this corpus are spelled correctly, simply collect all the words used to automatically create the dictionary and carry out a cleaning step. We exploited and downloaded various freely available sources of information (texts and Arabic dictionaries). Then, we cleaned the text and added to the dictionary: it is a file that will contain the maximum of words in Arabic [5].

In this section we describe the principle of detecting misspelled words, spell checking, and sorting of possible corrections. We used a simple approach to detect misspelled words based on the use of a dictionary: a word of text is considered misspelled when it does not appear in the dictionary. If the data volume is large enough, we can hope to cover enough cases to get a useful system. Each of the words of the text is compared to words in the lexicon.[1] Any word not found in the lexicon is considered wrong. To detect misspelled words, just browse the dictionary. We transliterated into Latin dictionary as a transliteration representing the Arabic characters as Latin characters because the handling of Arabic characters is difficult.

To browse the dictionary and detect errors we used the binary search because its use has been extremely efficient compared to conventional sequential search. Then we organized our dictionary as follows to make the quick search: we currently cut The dictionary into dictionaries following the length of each word ie words that have the same length are grouped in a dictionary with a noun that carries the length of the words that compose it.

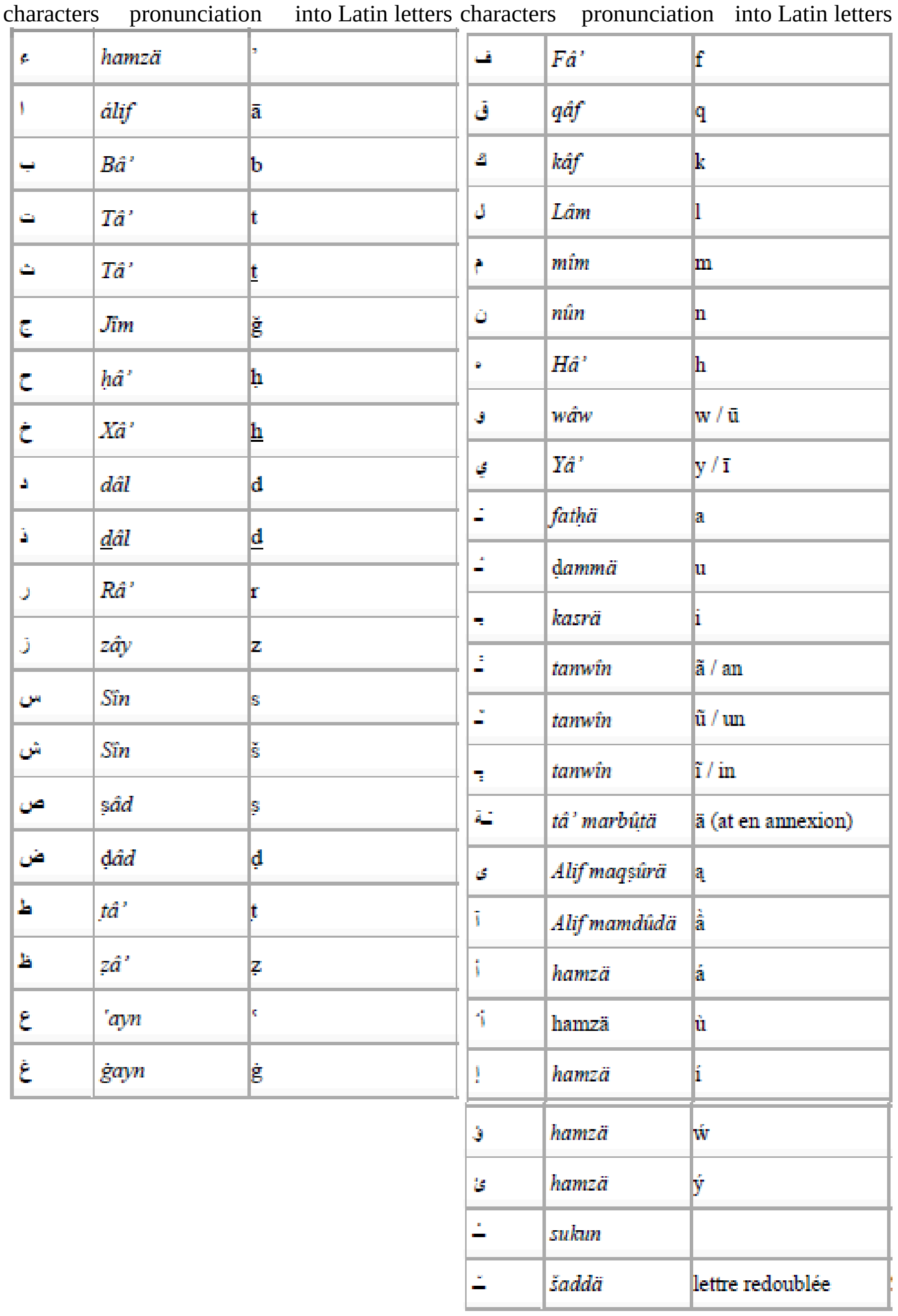
3.4 Spelling Correction: Use of Edit Distance to Order Suggestions

For a misspelled word we can add spaces in the middle of it to break it down into words. Our first prototype will analyze the words obtained and it will search in our data dictionary. If the words exist in the latter, they see them as a correction proposal.

The measurement of the best known distance[2], the Levenshtein distance is a simple metric between two channels, or each operation at a cost of 1 [7]. Levenshtein distance can be weighted by the length of the compared strings: the score is then divided by the sum of the lengths of the two compared strings. This is a distance in the mathematical sense, so in particular it is a positive number or zero, and two strings are identical if and only if their distance is zero. Calculate the minimum number of operations needed to transform one string characters in to another, where an operation is defined as the insertion, deletion, or substitution to move from one channel to another. One application of this distance is spellchecking: when a person types in a word, compared to a dictionary. If the word is present, nothing is done, otherwise, there are attempts from dictionary words [5], those whose Levenshtein distance to the typed word is less than a given limit. The nearest words are suggested as replacement first. The measurement of the Levenshtein distance between two strings (String1 and String2) consists of implementing the following algorithm:

[1] A glossary of animal words constituted the canonical forms (lemmas or bases), proclitics, prefixes, suffixes and enclitic, and a morphological analysis algorithm.
[2] http://en.wikipedia.org/wiki/Levenshtein_distance

Table 1. Transliteration of Arabic characters as Latin characters



Arabic characters	Characters: pronunciation	Transliteration into Latin letters	Arabic characters	Characters: pronunciation	Transliteration into Latin letters
ء	*hamzä*	ʾ	ف	*Fâʾ*	f
ا	*álif*	ā	ق	*qâf*	q
ب	*Bâʾ*	b	ك	*kâf*	k
ت	*Tâʾ*	t	ل	*Lâm*	l
ث	*Tâʾ*	ṯ	م	*mîm*	m
ج	*Jîm*	ğ	ن	*nûn*	n
ح	*ḥâʾ*	ḥ	ه	*Hâʾ*	h
خ	*Xâʾ*	ḫ	و	*wâw*	w / ū
د	*dâl*	d	ي	*Yâʾ*	y / ī
ذ	*ḏâl*	ḏ	ـَ	*fatḥä*	a
ر	*Râʾ*	r	ـُ	*ḍammä*	u
ز	*zây*	z	ـِ	*kasrä*	i
س	*Sîn*	s	ـً	*tanwîn*	ã / an
ش	*Sîn*	š	ـٌ	*tanwîn*	ũ / un
ص	*ṣâd*	ṣ	ـٍ	*tanwîn*	ĩ / in
ض	*ḍâd*	ḍ	ـة	*tâʾ marbûṭä*	ä (at en annexion)
ط	*ṭâʾ*	ṭ	ى	*Alif maqṣûrä*	ą
ظ	*ẓâʾ*	ẓ	آ	*Alif mamdûdä*	ȁ
ع	*ʿayn*	ʿ	أ	*hamzä*	á
غ	*ġayn*	ġ	أُ	hamzä	ù
			إ	*hamzä*	í
			ؤ	*hamzä*	ẃ
			ئ	*hamzä*	ý
			ـْ	*sukun*	
			ـّ	*šaddä*	lettre redoublée

Function 1.
Int LevenshteinDistance(char str1[1..lenStr1], char str2[1..lenStr2])

```
declare int d[0..lenStr1, 0..lenStr2]
declare int i, j, cost
for i from 0 to lenStr1
   d[i, 0] := i
for j from 0 to lenStr2
   d[0, j] := j
for i from 1 to lenStr1
   for j from 1 to lenStr2
      if str1[i] = str2[j] then cost := 0
      else cost := 1
      d[i, j] := minimum (
               d[i-1, j  ] + 1,        // deletion
               d[i  , j-1] + 1,        // insertion
               d[i-1, j-1] + cost      // substitution
      )
if (i > 1 and j > 1 and str1[i] = str2[j-1] and str1[i-1] = str2[j]) then
        d[i, j] := minimum (
              d[i, j],
              d[i-2, j-2] + 1          //  transposition
        )
return d[lenStr1, lenStr2]
```

Search word candidate for the correction is made with the edit distance reversed as follows: First, all words with an edit distance equal to 1 with the wrong password is generated by applying the editing operations that is the insertion, deletion, substitution and transposition. Then, each word is searched previously generated in e sorts or hash. If there is, then it is retained as a possible correction of the erroneous word.

Sorting candidates corrections takes into account the extent used in the selection step, as well as statistical measures. The choice of the most probable correction is done by giving each candidate a score Correction. The lower the score, the more likely it is that the candidate correction is the correct spelling of the word to correct.

Several ways to define this score are possible. The chosen solution is to set the score as the number of occurrences of the candidate correction in the text corpus more frequently correcting a candidate appears, it is more likely. This applies to sorting out words assigns a distance of levenshtein low enough to misspelled words.

4 Evaluation of the Prototype of Orthographic Correction

The corpus studied consists of a set of journalistic articles published by the newspaper "Le Monde Diplomatique" in its Arabic version. This source has the advantage of providing large quantities of good quality text The topics addressed are fairly general and treat various themes of the policy world news, economic, cultural, sports, etc. This corpus contains 1009 items, accumulating a total of 4,126,631 graphic words grouped under 322,156 different forms. Using a wide variety of themes and addressed areas aims to have a broad coverage of the words of the language. We used a corpus containing 164 Arabic texts collected in UTF-8 format.

The evaluation system is a crucial step. It helps to highlight its strengths and limitations, and to find leads for possible improvement. To get an accurate comparative assessment of our system, we chose different sizes of texts, each containing a varied set of misspelled words by comparing with other spell checkers nowadays, virtually present in all computer applications where the text is called to be entered by the user. This is usually notified of an incorrect entry with a red underline the wrong word. Examples of such applications are: the word processing software, email clients, source code editors and programming environments, internet search engines. The causes of the error are manifold and we find more than one way to classify [28].

Evaluation of any information retrieval system based on the calculation of a set of metrics. These calculations used to assess the proportion of errors displayed by the system from the ideal result. The metrics typically used are:

- **Number of words:** the number of words of each text.
- **Number of real errors**: the number of erroneous words of each text.
- **Number of errors detected**: the number of errors detected by the system.
- **Precision (P)** is an assessment of system noise. It measures the proportion of relevant system responses among all answers he provided
- **The recall (R)** is an assessment of the coverage of the system. It measures the amount of a relevant system compared to the number of responses ideal answers.
- **The F-measure (F)** is a metric that combines in a single value precision measurements and return to penalize excessive inequalities between the two measures.

However, given the boundary problems, we had to redefine the evaluation parameters to account for partially correct answers [32]. These valuation parameters become:

$$\textbf{Precision}: \boldsymbol{P} = \frac{\text{number of errors correctly (partially correct + incorrect) detected}}{\text{number of identified errors}}$$

$$\textbf{Recall}: \boldsymbol{R} = \frac{\text{number of errors correctly (partially correct + incorrect) detected}}{\text{actual number of errors}}$$

$$\textbf{F-measure}: \boldsymbol{F} = \frac{2.\mathrm{P}.\mathrm{R}}{\mathrm{P}+\mathrm{R}}$$

Table 2. Experience with our system.

Text	**Text 1**	**Text 2**	**Text 3**	**Text 4**	**Text 5**	**Text 6**
Number of words	100	80	60	70	50	40
In real errors Nb	33	27	23	17	13	11
Nb errors	23	20	19	12	11	9
Precision	69,69 %	74,07 %	82,60 %	70,58 %	84,61 %	81,81 %
Recall	71,23 %	83,37 %	81,49 %	78,14 %	85,75 %	80,41 %
F-measure	70,45 %	78,44%	82,04 %	74,16 %	85,17 %	81,10 %

Table 3. Experience with Word 2007

Text	**Text 1**	**Text 2**	**Text 3**	**Text 4**	**Text 5**	**Text 6**
Number of words	100	80	60	70	50	40
In real errors Nb	33	27	23	17	13	11
Nb errors	22	19	18	12	10	8
Precision	66,66 %	70,37 %	78,26 %	70,58 %	76,92 %	72,72 %
Recall	71,04 %	81,17 %	79,42 %	77,96 %	84,47 %	79,63 %
F-measure	68,78 %	75,38%	78,83 %	74,08 %	80,51 %	76,01 %

The three measures commonly used to assess a spelling correction system are the recall rate R, precision P and that of F. Measures to do this, and after the assumption of texts were aligned, the system begins its analysis sentence by sentence. The tests that are applied are encouraging. The summary evaluation we conducted shows that Arabic spellchecker gives in most proper words. The results of the evaluation show a gain of our method compared to the spellchecker "word 2007". After viewing these experiences, we see that these results are closer and are a good starting point for further research to improve our first spellchecking prototype.

5. Conclusion and Outlook

In this work we presented in detail the architecture of a first prototype for spelling Arabic. First, we describe the resources to spell checking and the implementation of this module. Then we evaluated our first prototype. A preliminary performance evaluation was conducted, which helped highlight the difficulty of the task and identify some of the current system limits. We can not certainly say that our work is complete when several improvements can be made. But we hope at least that we managed to achieve a simple spell check system.

As perspective of this work improvements can be made on the resources made increasing the dictionary size to cover a maximum of the Arabic language. Other techniques may be explored as ngram techniques and probabilistic. Semantic type errors can be a future project, too.

Referenes

1. Debili, F., Achour H., Souici E.: La langue arabe et l'ordinateur: de l'étiquetage grammatical à la voyellation automatique, Correspondances de 'IRMC, N°71, pp. 10-28, (2002).
2. Peterson, L.: Computer Programs for Detecting and Correcting Spelling Errors. Comm. ACM, 23 (1980).
3. Mitton, R.: Ordering the suggestions of a spellchecker without using context, Natural Language Engineering, (2009)
4. Kukich, K.: Techniques for Automatically Correcting Words in Text. R. Mitton, Ordering the suggestions of a spellchecker without using context, (2009).

5. Bacha, K., Zrigui, M.: Design of a Synthesizer and a Semantic Analyzer's Multi Arabic, for use in Computer Assisted Teaching, International Journal of Information Sciences and Application , IJISA , pp. 11-33, (2012).
6. Enguehard, C., Naroua, H.,: Evaluation of Virtual Keyboards for West -African Languages. Proceedings of the Sixth International Conference on Language Resource s and Evaluation (LREC'08) , Marrakech, Morocco, 28-30 (2008).
7. Levenshtein.V.: Binary codes capable of correcting deletions, insertions and reversals, (1966).
8. Kukich, K.: Techniques for automatically correcting words in text. ACM Computing Surveys, 24 (1992).
9. Peterson, J.: Computer Programs for Detecting and Correcting Spelling Errors. Comm. ACM, 23 (1980).
10. Pollock, K., Zamora, A.: Automatic Spelling Correction in Scientific and Scholarly Text, (1984).
11. Oflazer, K.: Error-tolerant Finite-state Recognition with Applications to Morphological Analysis and Spelling Correction, (1996).
12. Ben Othmane, Z., Zribi, A.: Algorithmes pour la correction orthographique en arabe, (1999).
13. Shaalan, K., Allam, A., Gomah, A.: Towards automatic spell checking for arabic. In LanguageEngineering, (2009).
14. Haddad, B., Yaseen, M.: Detection and correction of non-Words in Arabic: A hybrid approach. International Journal of Computer Processing of Oriental Languages, (2007).
15. Zerrouki, T., Balla, A.: Implementation of inxes and circum xes in the pellcheckers. In Proceedings of the Second International Conference on Arabic Language Resources and Tools, (2009).
16. Muaidi, H.: Extraction Of Arabic Word Roots: An Approach Based on Computational Model and Multi-Back propagation Neural Networks. PhDthesis, DeMontfort University-UK", (2012).
17. Gueddah.H., Yousfi, A., Belkasmi, M.,: Introduction of the weight edition errors in the Levenshtein distance, International Journal of Advanced Research in Artificial Intelligence, Vol 1 Issue 5, pp 30 32, Aout (2012).
18. Ndiaye, M., Vandeventer, F.,: A Correcteur orthographique apprentissage du franÁais. BULAG, 29:117-134 . (2004).
19. Levenshtein, V.: Binary codes capable of correcting deletions, insertions and reversals, (1966).
20. Yannakoudakis, E., Fawthrop, D.: The rules of spelling errors. Information Processing and Management, 19, 87-99 (1983).
21. Emirkanian, L., Louisette, N., Lorne, H.: La correction des erreurs d’orthographe d’usage dans un analyseur morpho-syntaxique du francais. Langue francaise, 106–122 (1989).
22. Heer, T. The application of the concept of homeosemy to natural language information retrieval. Information Processing and Management, 18(5), 229–236 (1982).
23. Angell, A., Richard C., George E.: Peter Auto-matic spelling correction using a trigram similarity measure. Information Processing and Management, 19(4), 255–261 (1983).

24. Lefevre P. Caillaud N., "Logiciel d'accès par voisinage à un dictionnaire automatique du français courant", (1992).
25. Sinah , R.: On partitioning a dictionary for visual text recognition, (1990).
26. Mazal, L., Vidal, E.: Computation of normalized edit distance and applications, (1993).
27. Stolcke, A.: SRILM An Extensible Language Modeling Toolkit. Proc. ntl. Conf. on Spoken Language Processing , (2002).
28. Merhbene, L., Zouaghi, A., Zrigui, M.: Arabic Word Sense Disambiguation.", ICAART (1) , 652-655 (2010)
29. Bacha, k., Zrigui, M.: Designing a Model of Arabic Derivation, for Use in Computer Assisted Teaching. KEOD (2012)
30. Bacha, K., Zrigui, M.: Morphological Analysis in the Environment "TELA". SCSE , 521-528 (2015).
31. Clarkson, M.: Statistical Language Modeling Using the CMU-Cambridge Toolkit.In Proceedings of EuroSpeech (1997).
32. Suzan, V.: Context-sensitive spell checking based on word trigram probabilities. Master thesis (2002).
33. Cunningham, H., Bontcheva, K.: Named Entity Recognition, Actes de la conférence internationale RANLP 2003, Borovets, Bulgaria, (2003).