A New Algorithm to Design and Implementation of Multilingual Spellchecker and Corrector

Dr. Hussein K. Alkhafaji

Al-Rafidain University College

dr.hkm1811@yahoo.com

Suhail A. Abdullah **Hanaa H. Merza**

suhail_almutair@yahoo.com hanaa.mirza@yahoo.com

Al-Rafidain University College

Abstract: *With the automatic or manual growths of text, the necessity of spell checking and correction have been increased. In spite of the availability of many algorithms for this purpose, there is no algorithm can be used as multilingual spell checker and corrector. Those algorithms are language–features oriented.*

This paper produced a new algorithm which depends on letters frequencies, weight of the letters and likeness measurements between the misspelled word and the candidates word.

This algorithm was used to design a spellchecker and corrector for Arabic, English, *French, Spanish, and German to prove its powerful implementation of multilingual spellchecker.*

Keywords : Multilingual, Type of error, Spell checking, Spell correcting.

1. Introduction

Despite the recent development in computer hardware and software, keyboard is the only way to input data; this gives a wide possibility of spell errors, because the keyboard is still suffered from lack of development. To overcome this problem, a lot of systems and algorithms have been designed for spell checking and correction for English, French, and German languages.

The languages of large number of letters and multi form letters and special characters like the Arabic language is still suffer from the problem of misspell. There are four types of errors, because of using keyboard for input the data as:

1- Two letters transposition.
2- One letter extra.
3- One letter wrong.
4- One letter missing.

Suppose that N is the number of letters of the examined word and L the number of letters in the language there are algorithms nominate the words to replace the examined word.

$$[(N-1) + (N) +L (N+1) +LN$$

When n is the length of the string and 26 is the size of an alphabet (L). So that a total of 53N+25 strings are checked against the dictionary if misspelled string is found. This number is driven from 26(N+1) possible insertions, N possible deletions, 25n possible substitutions and N-1 possible transpositions. [1] When N is number of letters of the misspell word so that the lengths of the nominated words are N, N-1 and N +1. Most algorithms used to design spell checkers and correctors depend on the order of a letter in the word, in which it takes a lot of time to find the correct word and might be impossible to find the correct word if the error is from type one (un neighbor letters) or other language of large number of letters or multi form letters like Arabic language. For example the misspelled English word "patertn" and intended word is "pattern", also the Arabic word "قماوس" and the intended word is "قاموس". Furthermore, the spell checking and correcting algorithms are language-oriented algorithms, i.e. they depend on the features of a language or group of related languages. Therefore they cannot be adopted to design multilingual spell checker and corrector systems. To exclude the above drawbacks and problems of spellchecking and correction algorithms, this research presents a new approach for spell checking and correcting. This approach depends on a weight for each letter or each form of a letter and similarity metric to measure the similarity between a word in a lexicon and the misspelled word under checking.

2. Related works

Dictionary lookup techniques and n-gram analysis are used to detect spelling errors. Dictionary up techniques are used to compare and locate an input string in a dictionary. These are exact string matching techniques with aim to reduce dictionary search time via efficient dictionary look up and/ or pattern – matching algorithms. If the string is not present in the chosen dictionary lexical or corpus, it is considered misspell word.

Probabilistic techniques often make use of language models constructed from n-grams. Combining the use of probabilistic techniques with dictionary look up techniques can achieve more desirable accuracy for languages such as Chinese with large alphabet and more complex structure.

Similarly key techniques are based on transforming words into keys that reflect their relation and characters. The words for which the keys are most similar are selected as suggestion. Such an approach is speed effective as only the words with similar keys have to be processed with a good transformation algorithm. This method can handle keyboard errors.[2]

Rule base techniques involve algorithm that attempt to request knowledge of common spelling error pattern for transforming misspelling words into correct ones. Each correct word generated by this process is taken as a correction suggest. The rules also have probabilities, making it possible to rank the suggestion by accumulating the probabilities for the applied rules. Edit distance can be viewed as special case of a rule-based method with limitation on the possible rule. Edit distance is useful for correcting errors resulting from keyboard input. It is not quite as good for correcting phonetic spelling errors, especially if the difference between spelling and pronunciation is big as in English or French.

Neural networks are also interesting and promising techniques which are based on back propagation networks using one output node for each word in the dictionary and an input node for every possible n-gram in every position of the word, where n is usually one or two. Normally one of the outputs should be active indicating which dictionary words the network suggests as a correction. This method works for small dictionaries, but does not scale well, the time requirements are to big on traditional, hardware, especially in the learning phase.[3]

N-gram base technique have been used as a basis or in combination with other spell correcting techniques in order to achieve the task of spell correction. N-gram does not show good performance on short words. N-gram similarity measure works best for insertion and deletion errors, well for substitution errors, but very poor for transposition errors.

A language-independent spell checker that is based on an enhancement of a pure n-gram model, furthermore, evaluations are done on English and Portuguese benchmark data set of misspell words. The proposed algorithm and data structure is not used to compute the similarity scores and not tested on data set for other languages specially. [4]

Statistical context base correction techniques make use of word bigram and trigram probabilities in order to capture collection trends. It requires large corpora and is computationally heavy. [5]

Spell correcting such languages tends to be context dependent, as the straight forward isolated-word error correction methods don't achieve high correction rate. Other issues such as word, text segmentation also provide a great challenge.

Editex is a mixture of soundex and edit distance. Though phonetic similarity is being taken in to consideration but other factor causing spell mistakes like keyboard is being still ignored. The performance is low hit rate and low precision.

The main techniques for error correction that have been explored: minimum edit distance is the most studied and used techniques for spelling correction to date. It is a technique where the minimum number of editing operations (insertions, deletions, substitutions, and transpositions) to transform one string into another.[6]

Edit distance between two strings is the minimum number of editing operations required to convert one string to the other. Edit distance is not sufficient; it needs to align each character of the two strings to each other. Again phonetic similarity is not being considered. [7]

Hamming distance and n-gram algorithms are integrated and have high recall for typing errors and phonetic spell checking algorithm in a single novel architecture. As these only shifts along the length of the input word, the shifting n-gram can be slow if shifted many places to the right. Hamming distance does not work well for insertion and deletion but works well for transposition and substitution. [8]

Spell checker for Arabic language based on dictionary look up and n-gram is achieved. For this purpose, eleven matrices are built to present the combination between the Arabic letters words. The accuracy is about 98-99%. Some problems faced the spell-checker like it may recognize many of incorrect words and consider them as correct. [9]

An Arabic diacritizer system is a baseline system , which is small in size, fast in processing and independent from other tools and linguistic. The system uses a statistical method that relies on quad-gram and probabilities, its accuracy rate is relatively high when compared to previous systems that are based only on statistics.[10]

In the next section, the algorithm presented in Fig-1 will be elucidated in details because it was adopted to design a multilingual spell checking. It presents the frequency of the letters of many languages and then driven weights depending on frequency tables.

3. The proposed algorithm

As it is mentioned previously most of spellchecking and correction algorithms are natural language-oriented algorithms, this leads to inability to design multilingual spell checker. Also, many languages contain many forms for same letter in their alphabets. All techniques have been failed to manipulate the misspell word of length greater than N+1 or less than N-1 and failed to correct words of transposition of un-neighbor letters.

The proposed algorithm is presented to overcome these drawbacks. It depends on simple strategy that is assigning a weight for each letter or a form of a letter. The weights of the letters and forms of the letters of specific language are driven from the frequency of its alphabet. To gain the goal of multilingual spell checker and corrector, this research was accomplished through three steps:

1) Special program is operated to count the frequencies of selected language for experiments.
2) Calculating the weights of the letters of a language under test.
3) Design a similarity function to measure the distance between the misspelled word and the word kept in the lexicon.

Tables 1, 2, and 3 show the frequencies of the letters of many languages. Table 1 illustrated the frequency distribution of the 26 most common letters across four languages (English, French, German, and Spanish) and some other special letters related to each language. Texts of more than 5000 letters are taken for each language to count the letters' frequencies and checked with the frequencies presented in [11] to achieve more reliability.

Table 1 Frequencies of 26 most common letters of 4 languages

Letters	English %(freq) 5060	German %(freq) 5915	Spanish % (freq) 5327	French %(freq) 5420
A	8.26(418)	6.4(379)	12.7(678)	7.3(395)
B	1.56(79)	2.5(148)	1.0(53)	0.8(43)
C	2.86(144)	3.6(213)	4.43(236)	3.53(191)
D	4.44(225)	4.94(292)	4.9(261)	4.6(249)
E	12.21(618)	15.0(887)	1602(863)	14.0(758)
F	2.15(109)	1.9(113)	0.62(33)	0.82(44)
G	2.0(101)	3.6(213)	1.01(53)	0.85(46)
H	6.14(311)	4.7(277)	0.3(16)	0.48(26)
I	7.07(358)	8.2(485)	6.7(357)	8.2(444)
J	0.2(10)	0.32(19)	0.28(15)	0.12(6)
K	0.8(40)	1.6(94)	0.03(2)	0.05(3)
L	3.91(198)	3.58(212)	5.05(269)	5.7(308)
M	2.51(127)	1.6(95)	3.3(177)	2.26(122)
N	6.64(336)	10.2(607)	6.9(367)	7.9(428)
O	7.19(364)	2.47(146)	8.6(457)	6.5(352)
P	1.86(94)	0.64(38)	2.25(120)	2.52(137)
Q	0.08(4)	0.017(1)	0.64(34)	0.83(45)
R	6.05(306)	7.11(427)	6.42(342)	6.5(353)
S	6.28(317)	8.2(485)	6.55(349)	7.53(408)
T	9.18(465)	4.9(289)	4.6(245)	9.0(487)
U	2.84(144)	3.65(216)	4.24(225)	5.2(282)
V	0.97(49)	1.64(97)	0.73(39)	0.94(50)
W	2.43(123)	1.55(92)	0.02(1)	0.12(65)
X	0.18(9)	0.05(3)	0.17(9)	0.35(19)
Y	1.88(95)	0.05(3)	0.17(9)	0.21(11)
Z	0.099(5)	0.8(47)	0.225(12)	0.14(7)

Table 2 Frequency of special letters related to some languages

Letters	Spanish % (freq) 5327	French %(freq) 5420	German %(freq) 5915
Π	0.58 (31)	-	-
Ñ	0.206 (11)	-	-
Ch	0.112 (6)	-	-
Û	-	0.02 (1)	-
É	-	2.68 (145)	-
À	-	0.654 (35)	-
Ê	-	0.234 (13)	-
̀e	-	0.104 (5)	-
̀u	-	0.04 (2)	-
Ô	-	0.02 (1)	-
Ü	-	-	0.794 (47)
ä	-	-	0.507 (30)
ö	-	-	0.304 (18)
ß	-	-	0.067 (4)

Table-2 presents the frequencies of uncommon letters in French, germen, and Spanish languages. The character "-", which is presented in the field of a specific language in the table means that the letter is not part of the alphabets of that language.

Table-3 presents the ratio of the frequency of Arabic letters and forms of letters. The highest letter frequency got the lowest weight while the lowest letter frequency got the highest weight. Different weight has been given to each form of multi forms letters. Therefore, the spell checker does not depend on the location or order of the letters. It depends on weight which has been driven from frequency of letters in texts. The weight is calculated according to eq.1

Weight(letter)=

10’s–complement(int(abs(log$_2$(frequency(letter))))).…eq.1

Table 3 Relative frequencies of letters in Arabic language

Letter	Percentage rate	Letter	Percentage rate
ء	0.95	س	2.37
أ	7.2	ش	0.6
ئ	0.6	ص	1.3
ؤ	0.36	ض	0.54
آ	1.3	ط	0.73
ا	12.46	ظ	0.36
ى	0.59	ع	2.92
ب	3.69	غ	0.63
ت	2.23	ف	1.5
ة	2.6	ق	1.7
ث	0.82	ك	2.0
ج	1.8	ل	12.5
ح	2.1	م	7.4
خ	1.4	ن	5.5
د	1.96	ه	2.1
ذ	0.73	و	5.97
ر	4.54	ي	6.02
ز	0.54		

Tables 4 and 5 show the weights of the letters depending on the letters' frequencies and eq.1.

Table 4 Weight of common and special letters in some languages

Weight	English	French	German	Spanish
1	E	E	e n	e a
2	a, i, o, t	s, t, u, a, i, l, o, r	a, d, h, i, r, s	d , i , l , n , o , r, s
3	h, n, r, s	c ,d ,p, é	b ,c ,g ,l ,o ,t ,u	c, m, t, u
4	d, f, l, w	M	f, k, m, v, w	P
5	c, g, m, p, u, y	f, g, q, v, w, à	z, ü	b, f, g, q, v
6	b, p, v	h, x	j, p, ä	Π
7	K	Y, ê	ö	h, j, z, ñ
8	x, j	J, z, ̀e	ß	x, y, ch
9	q, z	k, û, ̀u, ô	q, x, y	k, w
0	All non-alphabet letters			

Table-5 Weight of the Arabic letters

Weight	Letters
1	ا ل
2	أ م
3	ن و ي
4	ر
5	ب
6	ة ع
7	ت ج ح د س ف ق ك هـ
8	ء آ ى ث خ ذ ز ش ص ض ط غ ئ
9	ؤ ظ
0	All non-alphabet letters

The similarity function between the examined word and nominated word is counted by taking the total weight of the same letters which share both words.

The system removed all the nominated words which have differed in weight by more or less than nine. The words which have similarity percentage more than a user defined threshold will be a nominated word.

To make the nomination more accurate, likeness is counted according to eq.2

Likeness= similarity function – length of the nominated word……..eq.2.

The candidate word which has high likeness will be the nominated word.

```
1- Exclude all words that have length more than n-1 or n+1.
2- Exclude all words that have weight more than w-9 or w+9.
3- All words are not excluded by step 1 and 2 will be listed in a
   list called filtered list.
4- Similarity function = 0
5- Fetch one word from filtered list.
6- Count the total weight of letters which share in both words
   (nominated and examined).
7- If the percentage of similarity is more than a user defined
   threshold then the word will be nominated.
8- If there are more words in the list then go back to step 4.
9- Calculate the likeness for filtered list nominees.
10-Display all words which are nominated to the user to select
the desired word.
```

Fig-1 Similarity function algorithm

Step 1 and step 2 can be optioned for the designer. Their excluding increase the opportunity to find the correct word, (if it is stored in a lexicon), in spite of that the word includes more than one types of the errors. This property is not available in the rest algorithms. But, indeed, this will increase execution time required to check the filtered word list

The percentage of similarity in line 7 of fig-1 is user definer threshold. By experiments, if it is taken less than a user defined threshold, then the number of candidates will increase, and when the value is taken more than a user defined threshold, the chance of some candidate words will be less.

In the next section we count the Similarity function and the likeness between the examined word and the words in the dictionary for five languages. The increase of percentage will minimize the nominate words and the nearest words will emerge. The calculation of the similarity percentage is done according to eq.3

$$\textbf{Similarity percentage} = \frac{\textbf{total weights of examined word}}{\textbf{similarity function}} \ldots\text{..eq.3}$$

4. Examples for the Similarity Function Calculation

The method of similarity function and likeness calculation is as shown in Table- 6

Table- 6 Calculation of Similarity Function

Letters	**Weight**	**grace – grtcious**	**gracious – grtcious**
a	2		
b	5		
c	4	5	5
d	3		
e	1		
f	4		
g	4	5	5
h	3		
i	2		2
j	8		
k	7		
l	3		
m	4		
n	2		
o	2		2
p	4		
q	9		
r	3	3	3
s	3		3
t	2		
u	4		5
z	5		
w	4		
x	8		
y	4		
Z	9		
Total		13	25

Suppose that the misspell word "grtcuios" was compared with "grace" and "gracious", therefore the values of likeness functions are 13 and 25 respectively.

Likeness = similarity function – length of the nominated word

For grace = 13 – 5 = 8

For gracious= 25 – 8 = 17

Therefore the nominated word would be gracious, because of high likeness. Tables (7,8, 9,10, and 11) show the implementation of the same method above for different errors and different languages English, French, German, Spanish and Arabic respectively.

To explain the efficiency of the proposed methods, these examples manipulate misspell words which include one or more errors types with Length (misspelled) – Length (candidate)

Table- 7 Example of English language Similarity function

Type of error	**Nominated and examined Words**	**Similarity**	**Likeness**
One letter wrong	grace – grtcious	13	8
	gracious – grtcious	23	15
One letter missing	grace – gracius	15	10
	gracious – gracius	25	17
One letter extra	grace – gracious	15	10
	gracious – gracious	27	19
One letter transposition	grace – gracious	15	10
	gracious - gracious	27	19

Table-8 Example of French language Similarity function

Type of error	Nominate and examined words	Similarity	Likeness
One letter wrong	gracieuse–grecius	15	7
	gracieux–grecius	21	13
One letter missing	gracieuse–grcieux	15	6
	gracieux–grcieux	21	13
One letter extra	gracieuse – gracieuxe	17	8
	gracieux – gracieuxe	27	16
One letter transposition	gracieuse – grcaieux	17	8
	gracieux - grcaieux	23	15

Table- 9 Example of German language Similarity function

Type of error	Nominate and examined words	Similarity	Likeness
One letter wrong	grazie – grazie	13	7
	grazioes– grazie	13	5
One letter missing	grazie – graze	13	7
	grazioes – graze	13	5
One letter extra	grazie – graziee	16	10
	grazioes – graziee	16	8
One letter transposition	grazie – grazei	15	9
	grazioes – grazei	15	7

Table- 10 Example of Spanish language Similarity function

Type of error	Nominate and examined words	Similarity	Likeness
One letter wrong	agraciado – agraclado	17	8
	gracioso – agraclado	15	7
One letter missing	agraciado – agrciado	18	9
	gracioso – agrciado	16	8
One letter extra	agraciado – agraaciado	20	11
	gracioso – agraaciado	18	10
One letter transposition	agraciado – agracaido	19	10
	gracioso – agracaido	17	9

Table- 11 Example of Arabic language Similarity function

Type o error	Nominate and examined words	Similarity	Likeness
One letter wrong	حاسية – حاسبة	21	16
	حاسية – حسب	14	11
One letter missing	حسبة – حاسبة	25	20
	حسبة – حسب	19	16
One letter extra	حاسبية – حاسبة	26	21
	حاسبية – حسب	19	16
One letter transposition	حسابة – حاسبة	26	21
	حسابة – حسب	19	16

5. Conclusions

There are many conclusions emerged from this work, some of these conclusions:

1- In this paper, a new approach to design a multilingual spell checker and corrector. The experiments showed that the proposed algorithm ensures almost 100% finding the intended word, if the word is available in the dictionary.

2- A distinguished characteristic of this approach is its ability to handle the errors of two or more un-adjacent letters with 100% percentage if the intended word has same length with the candidate one.

3- The proposed technique has the ability to manipulate with 100% percentage of success the situation when the length of the misspelled word is N and the length of the candidates are greater than (N+1) or less than (N-1).

4- According to our survey, there is no method can handle, with 100% percentage of success, the situations where the misspelled word includes more than one type of errors. The proposed technique manipulated the situations where the word includes three types of errors; one or more letter wrong, one or more letter missing, and one or more letter transposition. Indeed, the situation required low level of user defined threshold.

5- Conclusions three and four required considerable amount of execution time to scan the dictionary to achieve 100% percentage of success.

6. Suggestions for Future works

We plan for the followings as future works:

1. To improve the efficiency of the dictionary scan, a morphological analyzer of a specified language can be added to the system and storing the roots of the words only in the dictionary.
2. The spell checker can be used in word processors, database systems, natural language interfaces, and official enterprises.

References

[1] Michel, A. A, "Spelling Corrector to Misspelled Names", comm. ACM vol. 30 no 3, 1987.

[2] Liang H. L., "Spell Checkers and Correctors: A Unified Treatment", M.Sc thesis, university of Pretoria, South Africa, 2008

[3] Gupta N. and Mathar P., "Spell Checking Techniques in NLP: A survey" International journal of advanced research in computer science and software Engineering, Vol. 2, Issue 12, Dec. 2012.

[4] Ahmad F., and Deluce E., "Revised N-gram Based Automatic Spelling Connection Tool to Improve Retrieval Effectiveness", Auemberger-polibits, 2009.

[5] Hodge v. j., "A comparison of Standard Spell Checking Algorithms", White rose Consortium, Universities Consortium (leeds, Sheffield, and York), 2003.

[6] Hussain S. and Nassem T., "Spell Checking", Crulp, Nuces, Pakistan, www.crulp.org

[7] Earnest Les, "The First Three Spell Checkers", Stanford University, retrieved 10 oct. 2010

[8] Hodge V .J. and Austin j., "A Comparison of Standard Spell Checking Algorithms and a Novel Binary Neural approach", IEEE transaction on knowledge and data engineering, pp. 1037-1081 vol. 15 no 5, oct.2003

[9] Muaidi H. and Tarawneh R.," Towards Arabic Spell Checker Based on N-gram scores", IJCA, vol. 53 no. 3, Sep 2012

[10] Alghamidi M., Muzaffer Z., and Alhakami H. ,"Automatic Restoration of Arabic Diacritics, A Simple Purely Statistical Approach", Arabian Journal, for Science and Engineering, 35(2c), 125-135, 2010.

[11] "Letter Frequency", Wikipedia, the free encyclopedia.

خوارزمية جديدة
لتصميم و تنجيز مدقق و مصحح إملائي متعدد اللغات

أ.م.د. حسين كيطان الخفاجي
كلية الرافدين الجامعة – قسم هندسة اتصالات الحاسوب
dr.hkm1811@yahoo.com

م. سهيل علي عبدالله
كلية الرافدين الجامعة
قسم علوم الحاسوب
suhail_almutair@yahoo.com

م.م. هناء حميد مرزا
كلية الرافدين الجامعة
قسم علوم هندسة البرامجيات
hanaa.mirza@yahoo.com

المستخلص:

إن النمو التلقائي أو اليدوي للنصوص، قد زاد من ضرورة التدقيق والتصحيح الإملائي. وعلى الرغم من توافر العديد من الخوارزميات لهذا الغرض، ليست هناك خوارزمية يمكن استخدامها للغات متعددة كون هذه الخوارزميات موجهة وفقاً لمميزات اللغة.هذا البحث يقدم خوارزمية جديدة تعتمد على ترددات الحروف وعلى وزن تلك الحروف وقياسات الشبه بين الكلمة الخاطئة والمرشحة. تم استخدام هذه الخوارزمية لتصميم المدقق الإملائي ومصحح للغة العربية والإنجليزية والفرنسية والإسبانية، والألمانية لإثبات قوة تنفيذه كمدقق إملائي للغات متعددة.