Adaption of Levenshtein algorithm for Albanian language

ALBAN RASHITI
ARBEN DAMONI
Faculty of Computer Science
Riinvest College
Lidhja e Prizrenit, No.42, Prishtina
KOSOVO
alban.rashiti@riinvest.net, arben.damoni@riinvest.net, http://www.riinvest.net

Abstract: - Levenshtein Algorithm also known as Levenshtein distance, is an algorithm which measures distance between words and also it is also used for conversion from one word to another. Measuring the distance between words is done by measuring the distance between the characters used in those words. Levenshtein algorithm works quite well for alphabets which in their composition have only letters consisting of a single character such as English language, but in cases where the alphabets have letters in their composition consisting of two or more characters such as Albanian language, this algorithm does not calculate the proper distance. Albanian language has 9 letters composed of two characters, namely dh, gj, sh, th, ll, rr, nj, xh and zh, therefore a new approach has been proposed for such cases.

Key-Words: - AL-Levenshtein, Characters, Data Processing, Levenshtein, Results, Time, Verification, Data

1 Introduction

Levenshtein algorithm [1] since its publication (1966) until now, has found applications in various systems and computer innovations, including correction and prediction systems for words, optical character recognition [2], search programs[3] and many other applications that are used daily[4][5][6]. Levenshtein algorithm works fine for all languages which in their alphabets have only letters composed from one character like English language. In languages which in their alphabets have letters composed from two or more characters, like Albanian language, Levenshtein algorithm does not calculate proper distance between letters. We have proposed an alternative approach whereby the theory is verified practically and has gone through the calibration stage initially, followed by three sets of data, consisting of more than fifty-five thousand words of Albanian language. The results have shown reduced Levenshtein distance, as well as the reduction in memory and processing time.

2 Levenshtein Algorithm

In Information theory and computer science, Levenshtein algorithm also known as "Edit Distance" or "Levenshtein Distance", is a metric measure which is used to measure difference between two words or strings [7]. In informal manner, Levenshtein distance between two words is the minimum of basic operations to transform one word to another. Edit basic operations are defined by Levenshtein in 1966 and these are: insertion, deletions and substitutions. If there are two string *a* and *b* in an alphabet Σ, for example an ASCII characters set or bytes set [0..255] etc., edit distance d(a, b) represents minimal number of operations which transform string *a* into string *b*.
Using Levenshtein approach there are different ways to calculate this distance. Most common is in table form as shown in table 1 below, which calculates distance between two string calculating every letter in strings.

		X_1	X_2	…	Xi	…
	0	1	2	3	4	5
Y_1	1					
Y_2	2					
…	3			$d_{i-1,\square-1}$	$d_{i,\square-1}$	
Y_i	4			$d_{i-1\square}$	$d_{i\square}$	
…	5					

Table 1. Measure of Levenshtein distance in table form [2]

In table 1, the first row X_1, X_2, … X_i represents letters for first string, while the first column Y_1, Y_2, …, Y_i represents letter for second string. Numbers from 0 to 5 represents distance which can increase for 1 for every different letter between two strings,

if letters are same in two strings, distance will not change (remains the same)[8].

2.1 Application of Levenshtein Algorithm

Levenshtein algorithm is applied in different systems and different fields and combined with other algorithms.
Levenshtein algorithm differs when applied to different languages because languages differ from each other. This difference between languages may be measured and is represented in the following table. Here the results of a study which accounts differences between some basic words of English language and several other languages, among them Albanian language. [9]

Language	Difference from English language
Swedish	63.88%
Danish	66.69%
Dutch	66.78%
French	69.31%
German	72.27%
Spanish	76.89%
Italian	82.14%
Albanian	88.61%
Croatian	90.74%
Estonian	91.45%
Polish	92.48%
Magyar	102.2%

Table 1. Calculation of the difference between the English language and some other languages [9]

2.2 Defining the problem of Levenshtein algorithm for Albanian Language

During the implementation of Levenshtein algorithm in word recognition and suggestions in Albanian languages and other languages that have letters in their alphabet composed of two characters, it has not offered optimal distance between letters.

Some of problems encountered during calculation with Levenshtein algorithm in Albanian language strings are:

- Distance for two character letters is larger
- In prediction and searching word applications these words are not ranked first
- Often words with two characters letters are not predicated accurate.

Therefore a new approach has been proposed.

There are other problems that occur with the use of Levenshtein algorithm in combination with other algorithms but that is not the scope of this paper.

3 AL-Levenshtein algorithm

After detailed review and implementation of Levenshtein algorithm, with various Albanian words, the result was larger distance compared to with English words.

Our proposed approach makes the replacement of alphabet letters composed of two characters with some single special characters from the ASCII that are not used in combination with other characters to form words. This novelty method makes the replacement before regular Levenshtein algorithm is used in calculating the distance.

The Levenshtein distance is implemented using C# programming language, with the added functionality through method called "*ReplaceLetters*", which makes replacement of two characters letters with one character from ASCII code, characters which are not in use in Albanian language. Replacement of two characters letters is shown in the table below.

DH	→	>
GJ	→	<
LL	→	#
NJ	→	@
RR	→	$
SH	→	^
TH	→	&
XH	→	*
ZH	→	~

Table 3. Replacement of two characters letters with ASCII symbols

4 Implementation results and their analysis

The modified Levenshtein algorithm, is named

AL-Levenshtein, because of its adaptation for Albanian language. Data used for implementation of AL-Levenshtein algorithm are explained in table 4.

USED DATA			
No.	**FIELD**	**USED WORDS**	**DATA SET**
1	Albanian names	214	First Set
2	Words with 2 characters	805	Second Set
3	Agriculture	4417	Third Set
4	Education	6162	
5	Economy	5736	
6	Legal	6492	
7	Culture	5999	
8	Literature	5718	
9	Politics	5156	
10	Health	5375	
11	Sport	5734	
12	Technology	4905	
TOTAL WORDS USED		**56713**	

Table 4. Used data during implementation

Data which are mentioned in table 4 are used in three different phases of calculation with Levenshtein and AL-Levenshtein algorithm. Calculation of these data is done in two ways, with manual implementation and automatic implementation. Manual implementation is when two character letters are changed manually in words whereas automatic implementation is when two character letters are changed automatically through programming in words. Data are applied in three sets. In the first set is composed from 214 Albanian names, and all of them have in their composition at least one two characters letters. On this dataset manual two character letter swapping is used and the comparison between two algorithms Levenshtein and AL-Levenshtein is provided in the figure 1 below.

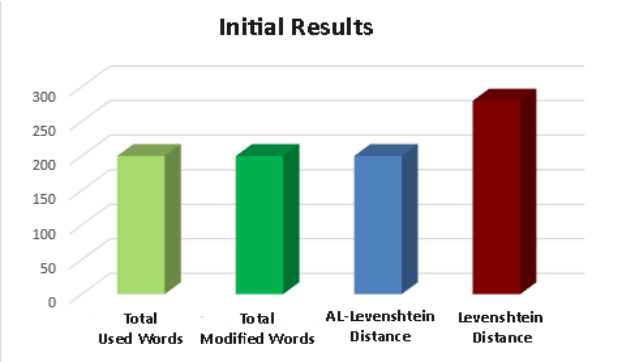

Figure 1. Results of initial phase implementation with Levenshtein algorithm and AL-Levenshtein.

In figure 1 with green are two pillars, one represents number of used words which are 214 used words and another represents modified words also are 214 modified words, while with blue is shown calculated distance with AL-Levenshtein which is the same distance as number of used words, it is 214 whereas pillar with red color is shown calculated distance with Levenshtein algorithm which is nearly 300 for 214 used words.

The second dataset is composed from 805 Albanian words and every word in this dataset contains at least one of the two character letters. The automatic and manual character swapping are used in this dataset and then the comparison between the Levenshtein and AL-Levenshetin are shown in figure 2 below.

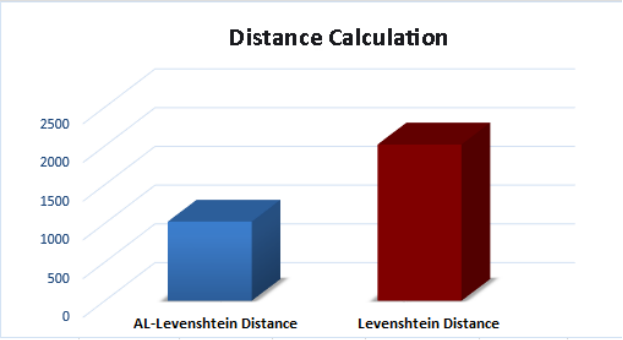

Figure 2. Distance calculation in second phase of implementation

As it can be seen the distance is twice as large with Levenshtein algorithm (red pillar) compared with AL-Levenshtein algorithm (blue pillar). Levenshtein distance reaches 1600, but AL-Levenshtein distance keeps the distance at 805 identical to the number of words used.

It is observed that there is 50% reduction in memory usage, as shown in figure 3.

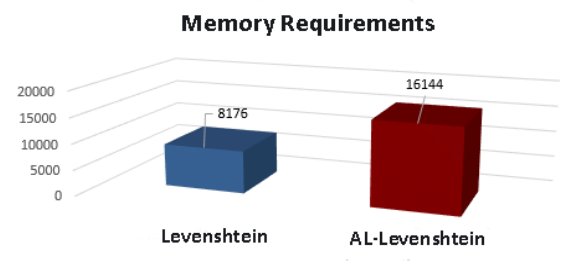

Figure 3. Memory requirements calculation for second phase.

Levenshtein algorithm, in red bar, is using 16144 bytes, while Al- Levenshtein represented in blue bar is using 8176 bytes.

This directly benefits the processing time which is reduced for AL-Levenshtein algorithm.

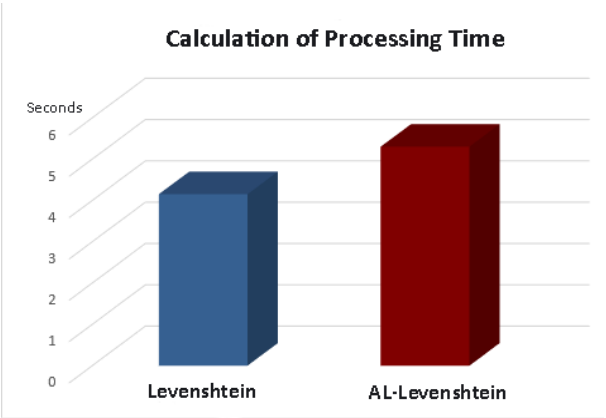

Figure 4. Calculation of the processing time for second phase

In figure 4 is shown the visual representation of the calculated processing time, where AL-Levenshtein algorithm, blue pillar, processes data in 3.8 seconds whereas Levenshtein algorithm, red pillar, processes the same data in 4.95 seconds.

The third dataset is composed from over fifty five thousand words from Albanian language from different fields such as: 1-Agriculture, 2-Education, 3-Economy, 4-Law, 5-Culture, 6-Literature, 7-Political, 8-Healthcare, 9-Sport and 10-Technology. The AL-Levenshtein calculations are done for distance, memory usage and processing time, and these are represented through figures 5, 6 and 7 respectively.

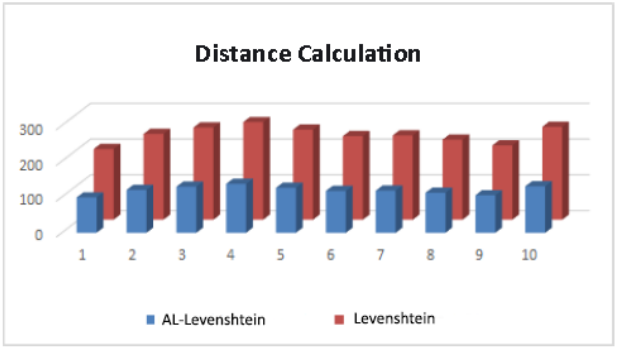

Figure 5. Distance calculation in third phase of implementation

Figure 5 shows the distance calculation comparison between Levenshtein vs AL-Levenshtein algorithms for ten different fields when used with third dataset. It is clearly seen that distance calculated with AL-Levenshtein it is much smaller than distance calculated with Levenshtein algorithm. Distance with AL-Levenshtein is more than 50% smaller than distance with Levenshtein Distance.

In figure 6 the memory used for the large dataset is represented for all the 10 fields individually.

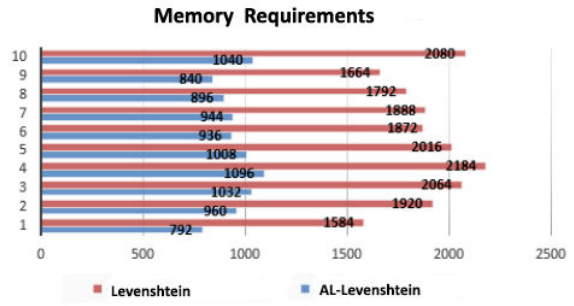

Figure 6. Memory requirements calculation for third phase

In blue are AL-Levenshtein and in red are Levenshtein algorithm memory usage results. Results show that memory requirements using AL-Levenshtein algorithm are on average 50% lower than with Levenshtein algorithm.

Results from the third benefit, reduced processing time, are represented in figure 7 below.

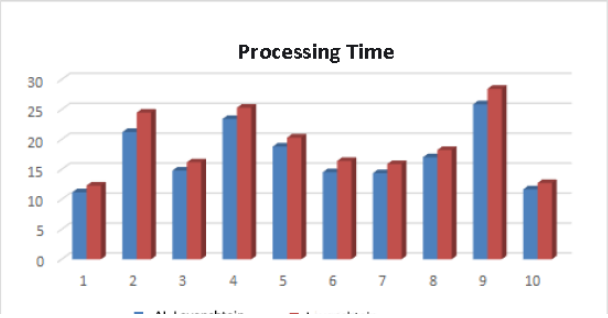

Figure 8 Calculation of the processing time for third phase.

Here the results are represented for each of the 10-fields with Levenshtein algorithm in red and AL-Levenshtein algorithm in blue. As it can be visually seen, processing time using the AL-Levenshtein algorithm is slightly smaller approximately 10% than with Levenshtein algorithm.

5 Conclusions

Levenshtein algorithm works well in different languages but can improve in those languages that in their alphabets have letters composed from two or more characters, among these languages is the Albanian language. To improve the distance calculation for Albanian language it is our recommendation to modify the algorithm that fits substitution of Albanian language two character letters and this modification produces AL-Levenshtein algorithm. With this modification three important results are achieved, namely distance between words is reduced up to 50%, memory usage is reduced by 50% and processing time is reduced for up to 10 %

References:

[1] Christopher D. Manning, Prabhakar Raghavan and Hinrich Schutze Edit Distance 2009: Stanford University.

[2] Filip Cristino, Iain D. Gilchrist and Jan Theeuwes - A simple way to estimate similarity between pairs of eye movement sequences 5(1):4, 1-15, Journal of Eye Movement Research.

[3] Riya Mary, A.Sayali Nishikant, C.Jaya Subalakshmi and N.Ch.N. Iyengar Use of Edit Distance Algorithm to search a Keyword in Cloud Environment Vol.7, No.6(2014), pp.223-232: International Journal of Database Theory and Application.

[4] Nimisha Singla, Deepak Garg String Matching Algorithms and their Applicability in various Applications ISSN:2231-2307,Volume-I, Issue-6, January 2012: International Journal of Soft Computing and Engineering (IJSCE).

[5] Microsoft Check Spelling and Grammar in Microsoft https://support.office.com/enUS/article/Check-spelling-and-grammar-cab319e8-17df-4b08-8c6b-b868dd2228d1.

[6] www.google.com Autocomplete Correction in Google,www.google.com

[7] Hinrich Schutze, "Introduction to Information RetrievalDictionaries and tolerant retrieval, 2014-04-10: Center for Information and Language Processing, University of Munich.

[8] Hinrich Schutze Levenshtein Distance 2014: Information and Language Processing, University of Munich.

[9] *http://ben.akrin.com/?p=728 – Distance from English Language*