Search Engine of Subject Using Error Correction Lavenshtein

(Case Study Digital Documents of Al Qu'ran and Hadith)

Edy Santoso[1], Marji[2], Nurul Hidayat[3]

[1,2,3]Studies Computer Science / Information Technology, University of Brawijaya

Abstract— Qur'an and Hadith is the holy book for Muslims as a way of life in everyday life. Qur'an itself consists of 30 juz, 114 letters and approximately 6,666 verses. While the Hadith is also very much making it difficult for Muslims who still lay preacher or a legal basis for the search for subjects interrelated among the verses in the Qur'an and Qur'anic verses linkages and Hadith. Today, existing search engines (Icon Find) but did not do a grouping of words that have been found so that the reader is difficult to understand because the scattered pages of documents. On the other hand people tend to have typing errors to look for a particular subject and if an error occurs writing the words that contain a particular subject was not found. Clerical errors are generally caused by the proximity of the keyboard layout, less adept at using finger, or because the two characters are located confused.
Levenshtein algorithm is an algorithm that is reliable and can be used to calculate the relationship between the strings by way of calculating the distance or amount of difference between two strings. With this method the system is expected to have mistyped the recommendations and improvements to search and classify the subject of the search results in a separate sheet that allows the reader.
Based on the experimental results generated that included the greater accuracy that the fewer the number of words recommended.

Keywords— Search Engines, Writing Errors, Levenshtein algorithm, The Qur'an and Hadith digital.

1 PRELIMINARY

Qur'an and Hadith is the holy book for Muslims as a way of life in everyday life. Qur'an itself consists of 30 juz, 114 Letters and approximately 6,666 verses. While the Hadith is also very much making it difficult for Muslims who still lay preacher or a legal basis for the search for subjects interrelated among the verses in the Qur'an and Qur'anic verses linkages and Hadith, especially if done manually by flipping per page. Today, existing search engines *(Icon Find)* but did not do a grouping of words that have been found so that the reader is difficult to understand because the scattered pages of documents. On the other hand people tend to have typing errors to look for a particular subject and if an error occurs writing the words that contain a particular subject was not found. Clerical errors are generally caused by the proximity of *the keyboard* layout, less adept at using finger, or because the two characters are located confused.

Search engines verses of the Qur'an has been studied by the U.S. Anwar, Abidin Z, Ririen Kusumawati R [3]. In that study a search technique using *exact string matching* technique, which is a technique that is in accordance with paragraph search input word correctly. The technique is most appropriate when the software user types a word or phrase to be searched by typing the word correctly or with no errors. But if the user types a word wrong in the input, the software does not provide solutions or possibilities of the verse in question.

Associated with the method, *Lavenshtein* algorithm has also been used to study Winoto, H [5] which states that the *Levenshtein Distance* algorithm is one of a string matching algorithm that can be used to detect plagiarism and has proven capable of detecting the similarity of text documents. This is characteristic of *the Levenshtein* algorithm calculates the relationship between the strings by way of calculating the distance or amount of difference between two strings. By selecting each of the distance between the two words, it can be used for a similar word recommendation. So with this method are expected to be used as a typing error and system recommendations for the improvement of the subject search and categorize search results in a

separate sheet. It is easier for the user.

2 BASIC THEORY

2.1 Search Engine

Search engine *(search engine)* is growing rapidly both search word in a document, search data in a database, also search word in the specific webpage. Search engines ayath of the Qur'an has been studied by the U.S. Anwar, Abidin Z, Ririen Kusumawati R [1]. In that study a search technique using *exact string matching* technique, which is a technique that is in accordance with paragraph search input word correctly. The technique is most appropriate when the software user types a word or phrase to search correctly. But if the user types a word wrong in the input, the software does not provide solutions or possibilities of the verse in question.This study also combines techniques *stemming* and *exact string matching* technique. *Stemming* is used as *preprocessing* for exact *string matching. Stemming* is used to find the base word of the affixes word by eliminating all the good affixes consisting of prefixes, suffixes, infixes but in this study only eliminates any prefixes and suffixes, for example if affixes word is "ordeal" the basic word is "try". *Exact string matching* is a matching string appropriately with the arrangement of characters in a string that has matched the number or sequence of characters The same, for example the word "try" to show compatibility only with the word "try". In connection with verse search, the results stemming will be used as keywords *(keyword)* search index of Qur'an database. The combination is intended to improve results of paragraph search and can further be categorized as an *inexact string matching* technique. The study produced the highest F-measure on the test data is 100% and the lowest *F-measure* is 66.66%. In that study the Arabic words have been found to be characterized without the sort of subject that exist in the Qur'an.

2.2 Lavenshtein Methods

In information theory, *the Levenshtein distance* of two strings is the minimum number of operations needed to transform one string into another string, where such operations are operations insertion, deletion, or substitution a character. This algorithm is named after *Vladimir Levenshtein* found in 1965. In this paper, the Lavenshein distance is referred to only in order use the word shorter distance. The basic is the determination of the distance of two strings can be formed through a recursive relationship. The calculation of the distance between two strings is the minimum number of operations is determined from changes to make the string A into string B. There are three main kinds of operations that can be performed by this algorithm are:

1. **Character Replacement Operation.** Character replacement operation is an operation to swap or replace a character with another character string for example, the authors write "*yamg*" to "*yang*". In this case the character "*m*" is replaced with the letter "*n*".
2. **Character Addition Operation.** Character addition operation is meant to add character into a string. For example, the string "*kepad*" into the string "*kepada*", the addition of the character "*a*" at the end of the string. The addition of a character not only performed at the end of the word, but can be added or inserted in the middle of the beginning of the string.
3. **Character Deletion Operation.** Character deletion operation is performed to remove characters from a string. For example, the string "*barur*" the last character string is removed so that it becomes "*baru*". In this operation the removal of the character "*r*" [1].

Description of Lavenshtein algorithm is as follows:

Base:

```
levDis ("", "") = 0
levDis (s, "") = levDis ("", s) = | s |
```

Based on this statement, there are two bases. The first line states clearly that the two empty strings do not have the distance, means to transform one string into another is not needed any surgery. The second line states that the distance between an empty string is not the empty string is equal to the length (number of characters) in the string is not empty.

Recurrent:

```
LevDis (s1 + c1, s2 + c2) =
      min ((levDis (s1, s2) +
      (if (c1 = c2) then
          0
      else endif)),
          (LevDis (s1 + c1, s2) + 1),
          (LevDis (s1, s2 + c2) + 1)
      )
```

Based on these algorithms when compared to the second string is not empty, i.e. both have `c1` and `c2` the last character then there are three alternatives to determine the distance / second string. First, if `c1` and `c2` equal, then `c1` and `c2` do not need to be exchanged levdis mean distance `(s1, s2) + 0`, if different means only a change of `c1` into `c2` course, means the distance `levdis (s1, s2) + 1`. In the case of this is performed substitution operation. Second, the deletion operation can also be done `c1` of `s1` and `s2 + c2` turn it into so that the distance to `levdis (s1, s2 + c2) + 1`. Thirdly, similar to the second insertion operation can be performed also on `s1 + c1 c2` that has been converted into so the distance is `levdis s2 (s1 + c1, c2) + 1`. These three alternatives, all changes to the existing possibilities, and of the three sought in which the least distance to the min function is looking for the minimum value of the three values [4].

In the implementation of this recursive algorithm, there are three times for every recurrent recursive call. This makes the algorithm becomes very slow and only good use on a string consisting of the characters a little. Therefore, further investigation showed that the distances `s1` and `s2` depend on the distance `s1` 'and `s2`' only where `s1` 'is shorter than `s1`, and `s2`' is shorter than `s2`. While the distance `s1` 'and `s2`' depends on the distance `s1` 'and `s2` "where both are shorter than the previous one. This suggests that dynamic programming techniques can be used. To calculate the distance without using a recursive process, use the matrix $(n + 1) \times (m + 1)$ where n is the length of the string `s1` and m is the length of `strings2`. Here are two strings to be used as an example:

```
RONALDINHO
ROLANDO
```

we see glance, the two strings have distance 6. Means to convert the string into `ROLANDO` `RONALDINHO` needed 6 operations, namely:

1. Substituting N with L
   `RONALDINHO` → `ROLALDINHO`
2. Substituting L to N
   `ROLALDINHO` → `ROLANDINHO`
3. Substituting I with O
   `ROLANDINHO` → `ROLANDONHO`

4. Removing O

   ROLANDONHO → ROLANDONH

5. Removing H

   ROLANDONH → ROLANDON

6. Removing N

   ROLANDON → ROLANDO

Lavenshtein algorithm has also been used to study Winoto, H [5] which states that the *Levenshtein Distance* algorithm is one of a string matching algorithm that can be used to detect plagiarism and has proven capable of detecting the similarity of text documents. This is evidenced by the acquisition rate of *similarity* is high on the document plagiarism.

3 Research Methods

At this stage it will be discussed methods, design, and measures that will be implemented in research.

Stages of the research described in the following steps:

1. Preparing documents with editing the arguments of Digital Quran and Maktabah Syamilah, which can be downloaded at http://maktabahsyamilah.com.
2. Designing Database containing information readings, meanings and descriptions.
3. Designing a search engine measures the error correction method Lavenshtein digital documents Qur'an and Hadith.
4. Store and display a list of records to be elected to html format based on topics that have been corrected.

In the early stages of a good collection of Arabic words that have been created with the Latin alphabet and word Indonesian translation of the Quran which is selected by the filtering method and then insert dictionary in the database using a DBMS.

As for the word search with typing errors by the user account is through the application of the method by comparing Levenstein said that input by the user with a list of dictionaries. When the user input is determined first word accuracy rate (0...100), where the greater the smaller the tolerance fault. For more general overview of the error correction systems can Lavenshtein method is illustrated in Figure 1.

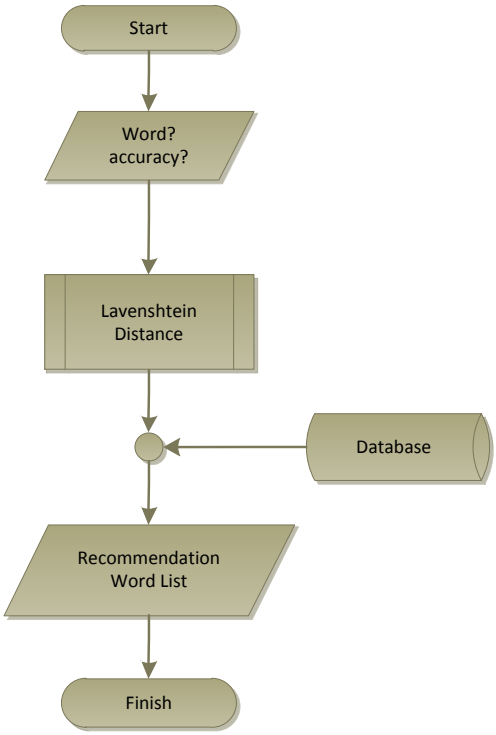

Figure 1. General description of the system

```
Function levDis (s1: string, s2: string): integer
Dictionary:
 i, j, cost: integer
 m: array [0 .. s1.length, 0 .. s2.length] of integer
Algorithm:
 for i = 0 to s1.length do
     for j = 0 to s2.length do
         if i = 0 then
              m [i, j] ← j          {comparison with the blank}
         else if j = 0 then
              m [i, j] ← i          {comparison with the blank}
         else                       {implementation of dynamic programming}
              if s1 [i] = s2 [j] then
                     cost ← 0
              else cost ← 1
                     m [i, j] = minimum (
                     m [i-1, j-1] + cost, {substitution}
                     m [i-1, j] + 1, {deletion}
                     m [i, j-1] + 1, {addition}
 )
 Return m [s1.length, s2.length]
```

For more details, Lavenshtein method steps can be shown in Figure 2.

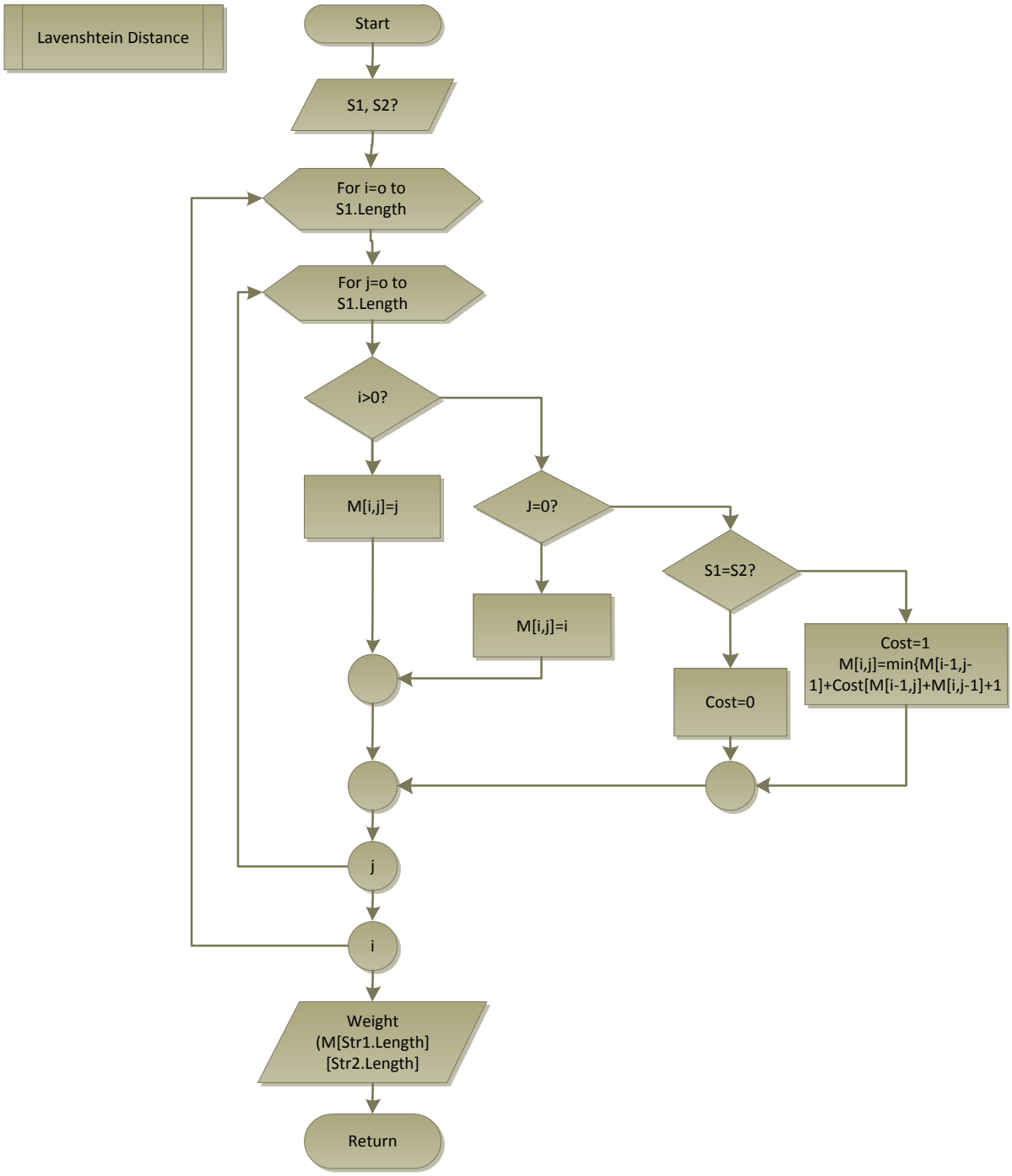

Figure 2. Flowchart method *Lavenshtein Distance*

4 RESULTS AND DISCUSSION

In this study, testing was conducted with three (3) additional treatment reduction, and replacement in accordance with the characteristics of said Lavenshtein method. For additional word done by giving the prefix, suffix insertion and addition. While the insertion is to add a character in the middle of the word and the addition of the suffix is added to the character at the end of the word. Table 1 and Table 2 is an example of testing for accuracy of 60% and an accuracy of 80%. Based on the table it can be seen the number of words found, that the words which have a distance close enough to the correct word.

TABEL 1. TEST RESULTS WITH 60% ACCURACY

No.	Words Correct	Increase		Reduction		Turnover	
		Word	Number	Word	Number	Word	Number
1	charity	amali	6	aml	1	amol	1
2	reading	bacaann	1	Bacan	11	bacaam	2
3	light	caahaya	1	Cahay	6	cahayu	0
4	degree	aderajat	2	deraja	6	derajah	3
5	you	enagkau	2	Engku	1	engkao	5
6	faith	pious	5	ima	2	imah	4
7	disposition	fitrahi	1	ftrah	1	pitrah	2
8	temptation	dgodaan	1	Godan	2	godaah	1
9	Forgiving	pemaafn	3	pemaf	5	pemaap	1
10	Alms	Zaakat	2	Zakt	7	Zakad	1
	On average 60%		**2.4**		**4.2**		**2.11**

TABEL 2. TEST RESULTS WITH 80% ACCURACY

No.	Words Correct	Increase		Reduction		Turnover	
		Word	Number	Word	Number	Word	Number
1	charity	amali	1	aml	0	amol	0
2	reading	bacaann	1	Bacan	1	bacaam	1
3	light	caahaya	1	Cahay	1	cahayu	0
4	degree	aderajat	1	deraja	1	derajah	0
5	you	enagkau	1	Engku	1	engkao	0
6	faith	pious	0	ima	0	imah	0
7	disposition	fitrahi	1	ftrah	1	pitrah	1
8	temptation	dgodaan	1	Godan	1	godaah	1
9	Forgiving	pemaafn	1	pemaf	1	pemaap	1
10	Alms	Zaakat	1	Zakt	1	Zakad	1
	On average 80%		**0.9**		**0.8**		**0.5**

Based on these tests can be performed for each recapitulation accuracy in detail is shown in Table 3.

TABEL 3. TEST RESULTS SUMMARY

No.	Accuracy	Increase	Reduction	Turnover
1	On average 50%	11.4	13.1	11.7
2	On average 60%	2.4	4.2	2.1111111
3	On average 70%	1	1.1	1.3
4	On average 80%	0.9	0.8	0.5
5	On average 90%	0	0	0
6	An average of 100%	0	0	0

Based on the graph is obtained that the greater accuracy of information desired by the user, the smaller the recommended word count and to an accuracy of 90% and 100% no words recommended by the system

when applied addition, insertion and reduction. However, if the written word is correct then produced the correct word itself.

Based on the research that has been done shows that the algorithms can be used for system *Lavenshtein* word search that has similarity with the correct word. For accuracy of 60% is applied to 10 (ten) said that true character is obtained by adding the average number of words found as many as 11.4 words, for the insertion of the word as much as 13.1 and 11.7 for the reduction of as much as words. As for the accuracy of 80% is applied to 10 (ten) said that true character is obtained by adding the average number of words found as much as 0.9 words, for the insertion of the word as much as 0.8 and 0.5 for a reduction of as much as words. In order to obtain the recommendation that the greater accuracy of the input the less said the recommended to the user, means actually approached said desired by the user. And conversely the smaller the accuracy of the input is also the greater number of words recommended by the system to the user, resulting in a user must act also choose some words that the system recommended. Recomendation of non-linear pattern (not a straight line), this is due to the distribution of words stored in the database with the proximity of each different word. Based on Table 3 is also known that for an accuracy of 90% to 100% there is no more words recommended by the system, it is because there is no distinct word except the word itself within an accuracy of 90% or 100%.

5 CONCLUSION

Based on these results it can be concluded that the method can be used to repair Lavensthein typing errors on word search system to recommendation subject search in the Qur'an and Hadith digital by considering improvements typing errors. The greater the degree of accuracy which included the fewer number of words generated recommendations or leads to a correct word.

6 BIBLIOGRAPHY

[1] Adriyani NM, Santiasa IW and Mulianttra A, "Implementation of the Levenshtein Distance Algorithm and Empirical Methods for Displaying Document Typing Errors Repair Advice Indonesian Language", Univ Udayana, Bali, 2011.

[2] Andhika, Fatardhi Rizky, "Application of the Algorithm String Suggestion Levenshtein Distance Algorithm and Other Alternatives in Applications", ITB, Bandung, 2010.

[3] Anwar U.S., Abidin Z, Ririen Kusumawati R, "Search Engines Using inexact Koran verse String Matching", Information Engineering, UIN Malang, 2011.

[4] Ilmy, MB., Rahmi, N., and Bu 'ulolo, RL, "Application of the Levenshtein Distance Algorithm for Correcting Spelling Errors in Text Editor", ITB, Bandung, 2006.

[5] Winoto, H, "Similarity Detection Algorithm Using the Text Document Content Levenshtein Distance", Information Eng., UIN Malang, 2012.