2020 Information Technology International Seminar (ITIS)
Surabaya, Indonesia, October 14-16, 2020

Levenshtein Distance Algorithm Analysis on Enrollment and Disposition of Letters Application

Sugiarto
Faculty of Computer Science
Universitas Pembangunan Nasional
"Veteran" Jawa Timur
Surabaya, Indonesia
sugiarto.if@upnjatim.ac.id

I Gede Susrama Mas Diyasa*)
Faculty of Computer Science
Universitas Pembangunan Nasional
"Veteran" Jawa Timur
Surabaya, Indonesia
igsusrama.if@upnjatim.ac.id
(corresponding Author)

Ilvi Nur Diana
Faculty of Computer Science
Universitas Pembangunan Nasional
"Veteran" Jawa Timur
Surabaya, Indonesia
ilvinur8@gmail.com

2020 6th Information Technology International Seminar (ITIS) | 978-1-7281-7726-7/20/$31.00 ©2020 IEEE | DOI: 10.1109/ITIS50118.2020.9321030

***Abstract*— Archives are recorded records of various activities or organizations in various forms, such as letters, books, and others, stored systematically in the place provided to facilitate searches if needed again. Archived data is recorded based on date, letter number, and other things related to digital filing. Digital filing of letters by storing softcopy documents, accompanied by incoming and outgoing mail reports, and monitoring mail disposition. To make it easier for agencies to process correspondence and letter archiving and speed up searching for letters, the Levenshtein distance algorithm is used. This research results in the form of a letter filing and disposition application that is easy in managing letters such as letter filing, letter disposing, and others by applying the Levenshtein distance algorithm in letter search to speed up the letter search process.**

Keywords*—*Levenshtein Distance, Analysis, Disposition, Enrollment, Letter.

I. Introduction

Archives are recorded records of various activities or organizations in various forms, such as letters, books, and others, stored systematically in the place provided to facilitate searches if needed again [1]. The archived data is recorded based on the date, letter number, and other matters relating to the archiving [1]. A letter is a communication tool to convey written information from one party and is addressed to another party to convey information. Letters are very closely related to agencies or institutions. Each day the agency will handle many letters. If the number of letters is not handled correctly, it will be detrimental to many parties. The archiving unit has a function that is to handle the receipt of incoming mail. An outgoing letter is a letter made by an agency to be sent to other agencies of an official nature [2].

In Indonesia, some agencies or institutions generally use conventional archive storage, causing large volumes of archives that can cause problems related to storage, maintenance costs, facilities, or other factors that can damage these archives. In general, this method cannot store for an extended period because conventional storage can accumulate files and damage the archives [3]. So that conventional storage is not very effective and efficient. And as technology develops, digital archive storage becomes the right solution for archive storage. However, the large amount of digital mail storage can cause problems when it comes to mail search. One alternative in minimizing the problems that arise in letter search is using the Levenshtein Distance algorithm because this algorithm can speed up searching for letters based on the words entered [4]. Data search with this algorithm is done by calculating the minimum number of operations required to convert a string into another string where an operation involves inserting, deleting, and replacing a single character [5].

Research by Yulianto et al. (2019), which is about Automatic Completion and Spelling Checking of the Distance Levenshtein Algorithm to Get Errors and Search for Data Texts in the Library, in his research using 1155 books in the Library, and the results were 1055 which were good [6].

Meanwhile, the research conducted by Rani et al. (2018) used the Distance Levenshtein Algorithm to check the level of document similarity (similarity) [7], as well as research conducted by Nurhayati (2017) made an android application to check the level of document similarity (similarity) with Using the Levenshtein Distance Algorithm. However, this study focuses only on evaluating document similarity checks without assessing or evaluating the level of plagiarism [8]. The results show the similarity document with the table of similarity check. Another study conducted by Lhoussain (2015) was to correct spelling in Arabic letters, but in testing, this study was limited because of the corpus' size [9].

From several previous studies, this paper discusses the Distance Levenshtein Algorithm analysis, which is applied to letter archiving applications by converting keywords in the form of an array, then selecting all the words in the keyword table, calculating distances, and taking the minimum value.

II. Methods

A. Levenshtein distance

Levenshtein Distance [10] [11] is a string matrix used to measure the difference between two strings. The distance value between two strings is determined by the minimum number of change operations required to transform from one string to another. These operations are insertion, deletion, and substitution. Mathematically, the value of the Levenshtein distance between two strings a, b, can be seen in equation 1. 1(ai≠bj) will be 1 if ai≠bj. Lev a,b(i,j), is the distance between the first 1 characters of a and the first j characters of b. [12]

There are three types of operations on the Levenshtein distance algorithm that can be done, including Character

Insertion (in section) operation. An operation that inserts characters into a string. For example, the string "distrit" becomes the string "discrete," and we insert the character "k" in the middle. String. Insertion is done in the middle of the string and inserted at the beginning or end of the string—an illustration like Table I [13].

$$\text{Lev a,b } (|a|, |b|)$$

$$\text{Lev a,b (I,j)} = \begin{cases} \max(i,j) \\ \min \begin{cases} lev\ a,b(i-1,j)+1 \\ lev\ a,b(i,j-1)+1 \\ lev\ a,b(i-j)+1(a \neq b\ j) \end{cases} \\ if \min(i,j) = 0 \end{cases} \quad (1)$$

TABLE I. INSERTION OF CHARACTERS

String 1	d	i	s	K	r	i	t
String 2	d	i	s	-	r	i	t
insection				K			

Character Erase Operation (Deletion) [14]. The operation removes characters from a string. For example, in the string 'informatikan,' the last character is dropped to become the string 'informatika.' This operation removes the 'n' character—an illustration like Table II.

TABLE II. DELETION OF CHARACTERS

String 1	i	n	f	o	r	M	a	t	i	k	a	-
String 2	i	n	f	o	r	M	a	t	i	k	a	n
Deletion												n

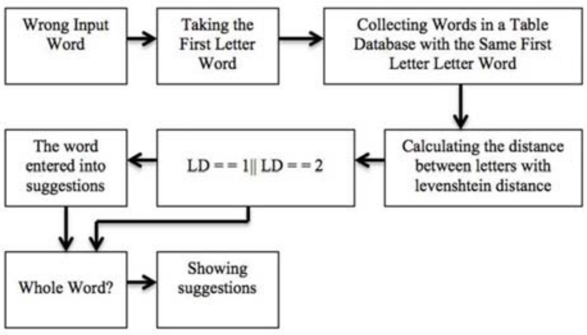
Character Exchange Operation (Substitution) The operation exchanges a character for another character. For example, the author writes the string "reknik" to "teknik." In this operation, the character "r" at the beginning of the string is replaced by the letter "t". An illustration like Table III.

TABLE III. SUBSTITUTION OF CHARACTERS

String 1	t	e	k	n	i	k
String 2	r	e	k	n	i	k
substitution	t					

B. Design System

The process of running the Levenshtein distance algorithm starts with a wrong word entered by the user, and then the first character is taken from the input in the search box. After that, the words in the database whose first characters or letters are the same are collected, and the distance is calculated. Then compare whether LD 1 (word input by the user) and LD 2 (word in the database) are the same or not? If the same, then the word will be included in the suggestions that will be raised. However, if it is different, then this step will lead to the next step, namely whether all the words appear or not? It will return to the initial step, namely finding the distance with the Levenshtein distance algorithm. But if what appears is the whole world, it will display a suggestion or prediction. Here the process stops. A shorter period is shown in Fig. 1.



Wrong Input Word
Taking the First Letter Word
Collecting Words in a Table Database with the Same First Letter Letter Word
Calculating the distance between letters with levenshtein distance
LD = = 1|| LD = = 2
The word entered into suggestions
Whole Word?
Showing suggestions

Fig. 1. Design System.

C. Letter Search Testing Process, with the Levenshtein distance algorithm

The Levenshtein distance algorithm [15] is a string matrix used to measure the difference between two strings. When searching for data on the system using keywords that are not by the spelling, the system will provide keyword suggestions, which result from correcting the spelling of the previous keywords. For example, the system suggests the keyword "dan" which is the correct spelling of "den". The spelling process as follows:

1. Convert keywords into arrays.
2. Make a selection of all the words stored in the keyword table.

```
public boolean onQueryTextSubmit(String query) {
        return false;
    }

    @Override
    public boolean onQueryTextChange(String nextText) {
        ArrayList<Suratmasuks> suratmasuks = new ArrayList<>();
        ArrayList<Integer> dataFilter = new ArrayList<>();
        if (nextText.isEmpty()) {
            suratmasuks.addAll(listDataSuratMasuk);
            sekre_suratMasukAdapter.setSMList(suratmasuks);
        } else {
            for (int i = 0; i < listDataSuratMasuk.size(); i++) {
                Suratmasuks datasurat = listDataSuratMasuk.get(i);
                dataFilter.add(distance(datasurat.getIsi(), nextText));
            }
            Collections.sort(dataFilter);
            for (int i = 0; i < listDataSuratMasuk.size(); i++) {
                Suratmasuks datasurat = listDataSuratMasuk.get(i);
                if (distance(datasurat.getIsi(), nextText) == dataFilter.get(0)) {
                    suratmasuks.add(datasurat);
                    sekre_suratMasukAdapter.setSMList(suratmasuks);
                }

            }
        }
        return false;
    }
```

3. Perform the distance calculation for the keyword "and" with each word selected in the keyword table. For example, a keyword is chosen as a comparison, namely "den" with a length of 3 characters.

4. The calculation method is to take the minimum value of d [i-1, j] + 1, d [i, j-1] + 1, d [i-1, j-1] + cost if i = 1 and j = 1 then d [1-1,1] + 1 = 1, d [1,1-1] + 1 = 2, d [1-1,1-1] + 0 = 0. Cost = 0 is obtained from if source string length = target string length. Meanwhile, cost = 1 if the length of the source string! = The length of the target string. The results of the calculation of distance values can be seen in Table IV.

```
private static int distance(String a, String b) {
    a = a.toLowerCase();
    b = b.toLowerCase();
    int[] costs = new int[b.length() + 1];
    for (int j = 0; j < costs.length; j++)
        costs[j] = j;
    for (int i = 1; i <= a.length(); i++)
    {
        costs[0] = i;
        int nw = i - 1;
        for (int j = 1; j <= b.length(); j++)
        {
            int cj = Math.min(1 + Math.min(costs[j], costs[j - 1]),
                    a.charAt(i - 1) == b.charAt(j - 1) ? nw : nw + 1);
            nw = costs[j];
            costs[j] = cj;
        }
    }
    return costs[b.length()];
}
```

TABLE IV. CALCULATING THE DISTANCE BETWEEN WORDS "DAN" WITH THE WORD "DEN"

		D	A	N
	0	1	2	3
D	1	0	1	2
E	2	1	1	2
N	3	2	2	2

An example of applying the Levenshtein distance algorithm to the application is when searching for letters. For example, when you want to search for the word "dana" but only write it down with the word "dan". Then, the system will provide keyword suggestions to provide spelling correction results with minimum word spacing. In the word "and" there are word suggestions, as shown in Fig. 2. From the results of the calculation of the matrix, the distance value for the words being compared is 1.

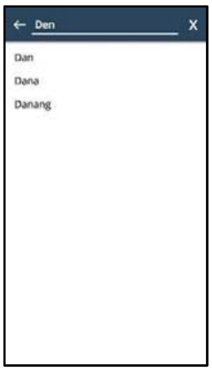
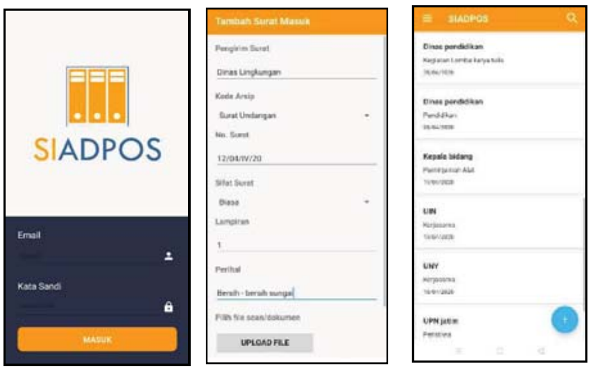


Fig. 2. An example of applying the Levenshtein distance algorithm in the application.

III. Experiments And Result

A. *Results of Trial of Application "Enrollment and Disposition of Letters"*

The application "Enrollment and Disposition of Letter" is an android application to help process correspondence such as incoming mail, outgoing mail, archive, and disposition of letters; by applying the Levenshtein distance algorithm, the process of searching for letters that have been stored in the database. Fig. 3 is a view of the process of " Enrollment and Disposition of Letter".



Pengirim Surat
Dinas Lingkungan
Kode Arsip
Surat Undangan
No. Surat
12/04/IV/20
Sifat Surat
Biasa
Lampiran
1
Perihal
Bersih - bersih sungai
Pilih file scan/dokumen

Dinas pendidikan
Kegiatan Lomba karya tulis

Dinas pendidikan
Pendidikan

Kepala bidang
Peminjaman Alat

UIN
Kerjasama

UNY
Kerjasama

UPN jatim
Peresmian

Fig. 3. (a) Login display, (b) Incoming mail display, (c) add incoming mail display.

Testing this application is carried out using the black box method, a test that only checks the application's functionality and observes the results of execution through data testing [16] [17]. In addition to black-box testing, comparison testing is also carried out between using algorithms and without using algorithms.

Testing is done by creating a questionnaire related to the system created and distributed to users related to the system assessment. Tabulation of data obtained from the results of a questionnaire that has been filled in by the user via Google form with ten questions related to the system. From the table, 36 respondents have filled out the questionnaire with the answer "Disagree" with a value of 1, "Disagree" with a value of 2, "Enough" with a value of 3, "Agree" with a value of 4, and "Strongly Agree" with a value of 5. The validity results are in the form of a questionnaire data correlation table, which then the Pearson.

Correlation value on the validity results is compared with the table r with the number of respondents 36-2 = 34. According to the distribution of values, the value of the table r 34 with a significance of 0.05 is 0.2785. Based on the analysis results, the Pearson Correlation value obtained from 10 items is more significant than 0.2785, which means valid. After getting the results of each valid item based on the validity test, the next step is to do a reliability test using the Cronbach's Alpha method [18] [19], which aims to determine the consistency value of the questionnaires that have been distributed so that the questionnaire is accurate.

The questionnaire was declared reliable by the criteria if Cronbach's Alpha was more significant than 0.60. From the analysis, the Cronbach's Alpha value obtained is 0.831, which means reliable.

The next test is to use Cohen's Kappa test. The first step is two different testers or respondents testing the application's functions according to the test scenario. For testing the functions in the application, the value is one of the function is running as expected. Value 2 if the function does not work as expected, there are 40 test scenarios and two respondents or examiners. The first examiner is carried out by author one, and the second examiner is carried out by author 2. From these results, then Cohen's Kappa test is carried out to determine the level of equality between respondents. According to [20][21], the level of perception equality is said to be almost perfect if the kappa value is in the range of 0.6-0.8. Cohen's Kappa test was conducted using IBM SPSS (Table V) so that the kappa coefficient value was 0.640. This means that the value of the test agreement between examiner one and examiner 2 is almost perfect.

TABLE V. TEST RESULTS FOR COHEN'S KAPPA

Symmetric Measures					
		Value	Asymptotic Standard Error	Approximate T^b	Approximate Significance
Measure of Agreement	Kappa	0.640	.236	4.045	.000
N of Valid Cases		40			

B. *Testing the Levenshtein distance algorithm*

Testing the Levenshtein distance algorithm by comparing the process of looking for letters in the application "letter archiving and disposition", using the Levenshtein distance algorithm and not using the Levenshtein distance algorithm.

- The letter search does not use an algorithm. In this test, it is done by entering words in a letter search. For example, searching for incoming mail about "kebersihan" by writing the word "kbrsihan" in the search results can be seen in Fig. 4.a. From Fig. 4.a., the search results are not found / not found. This is because there are errors in writing the word in the search process so that the results cannot be found.

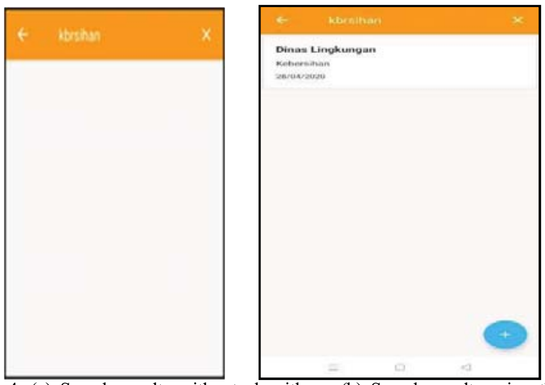


Fig. 4. (a) Search results without algorithms, (b) Search results using the levenshtein distance algorithm

- Search letters with the Levenshtein distance algorithm. In this test, searching for letters using the Levenshtein distance algorithm goes through several processes, namely inputting words in the letter search. The word is obtained from the subject of the letter that has been stored in the API, then the word that is entered is calculated the distance of the word that is close to the desired word, to display the word being searched for in the letter search. For example, searching for incoming mail about "cleanliness" by writing the word "kbrsihan" in the search results of this search can be seen in Fig. 4.b. the results of these searches are following the desired word, namely "cleanliness". Unlike the letter search without using an algorithm that does not find search results when inputting the word "kbrsihan". This is because the Levenshtein distance algorithm calculates the distance of the words inputted with the words that are close to the desired word.

The results are compared testing without using an algorithm and testing using the Levenshtein distance algorithm, shown in Table VI.

TABLE VI. SEARCH TEST RESULTS USING THE LEVENSHTEIN DISTANCE ALGORITHM

Input Word	The desired result	**Test Results**	
		without the algorithm **Levenshtein Distance**	**With Algoritma Levenshtein Distance**
Kbrsihan	Kebersihan	Shows no results	By the desired result
Kerjamasa	Kerjasama	Shows no results	By the desired result
Peristiw	Persistiwa	Shows no results	By the desired result
Pemijaman alat	Peminjaman alat	Shows no results	By the desired result
Perisahaan abcd	Perusahaan abcde	Shows no results	By the desired result
Perjinan	Perijinan	Shows no results	By the desired result
Jb fairr	Job fairrr	Shows no results	By the desired result

IV. CONCLUSION

In searching for letters, the Levenshtein distance algorithm is implemented by using several processes, namely inputting words in the letter search. The word is obtained from the subject of the letter that has been stored in the API, and then the word input has calculated the distance of the word that is close to the desired word to display the word being searched for in the letter search. Comparison of letter searches without using algorithms and letter searches using the Levenshtein distance algorithm in Archiving design and letter disposition can display data by the word being searched even though there are some errors when writing the word you want to search for, such as writing the word "kbersihan" the result that appears from the process is "kebersihan".

REFERENCES

[1] S.P. Airlangga, Teguh, M.C.R. Nugraha, "Archival Management at the University of Lampung", Fiat Justisia Journal, Vol. 12 (4), pp. 298-306, 2018.

[2] Durinda Puspasari and Choirul Nikmah, "Effectiveness of Archive Management on Record System in National Zakat Agency in Indonesia", *Advances in Social Science, Education and Humanities Research* – Atlantis Press, volume 222, pp. 283-288, 2018

[3] M. Churiyah, M. Arief, A. Basuki, B. A. Dharma and A. wulandari, "Archive Management in the Digital Age: Development of Village Administration Systems", *Advances in Economics, Business and Management Research-Atlantis Press,* volume 124, pp. 626-632, 2019

[4] M. Zhou, L. Wang and J. Chen, "Design of the Enrollment Information System for Foreign Students Under the Background of International

Education", 2nd International Conference on Manufacturing Science and Information Engineering (ICMSIE), pp. 29-38, 2017

[5] E. Pratama, L. A. Abdillah dan S. D. Purnamasar, "Correspondence Archival Information Systems In Bina Darma University", Proc. International Conference on Information Technology and Engineering Application (ICIBA), pp. 208-213, 2015.

[6] M. M. Yulianto, R. Arifudin, and Alamsyah, "Autocomplete and Spell Checking Levenshtein Distance Algorithm to Getting Text Suggest Error Data Searching in Library", *Scientific Journal of Informatics*, Vol. 5, (1), pp. 67-75, 2018

[7] S. Rani and J. Singh, "Enhancing Levenshtein's Edit Distance Algorithm for Evaluating Document Similarity", International Conference on Computing, Analytics and Networks-ICAN, Computing, Analytics and Networks, pp. 72-80, 2017.

[8] Nurhayati and Busman, "Development of document plagiarism detection software using levensthein distance algorithm on Android smartphone", International Conference on Cyber and IT Service Management (CITSM)-IEEE Xplore, Vol. 7316249, pp. 1-6, 2017

[9] A. S. Lhoussain, "Adaptating The Levenshtein Distance To Contextual Spelling Correction", *International Journal of Computer Science and Applications*, Technomathematics Research Foundation, Vol. 12, No. 1, pp. 127-133, 2015

[10] B. D'Cunha, R. K. Karkada, and S. Vijaykumar, "A Comparative Analysis Of String Metric Algorithms**",** *International Journal of Latest Trends in Engineering and Technology* Special Issue SACAIM, pp. 138-142, 2017

[11] Z. Wen, D. Deng, R. Zhang, and K. Ramamohanarao, "2ED: An Efficient Entity Extraction Algorithm using Two-Level Edit-Distance", 35th International Conference on Data Engineering (ICDE), IEEE-Xplore, 2019, pp. 998-1009

[12] G. Veena and G. alaja, "Levenshtein Distance based Information Retrieval", *International Journal of Scientific & Engineering Research*, Vol. 6, Issue 5, pp. 112-116, 2015.

[13] Y. M. Myint, "Edit distance computation with minimum number of edit operations in database management system and information retrieval", *International Journal of Scientific and Research Publications*, Vol. 9 (9), pp. 651-656, 2019

[14] H. N. Abdulkhudhur, I. Q. Habeeb, Y. Yusof, and S. A. M. Yusof, "Implementation Of Improved Levenshtein Algorithm For Spelling Correction Word Candidate List Generation", *Journal Of Theoretical And Applied Information Technology*, Vol.88. No.3, Pp. 449-455, 2016

[15] E. Alexandru and E. Andre, "An application of Levenshtein algorithm in vocabulary learning", International Conference –Electronics, Computers and Artificial Intelligence (ECAI)-IEEE Xplore, pp. 1-4, 2017.

[16] M. N. W. Khairun Nisa and H. Wibawanto, "Implementation of the Autocomplete Feature and the Levenshtein Distance Algorithm to Increase the Effectiveness of Word Search in the Big Indonesian Dictionary (Implementasi Fitur Autocomplete dan Algoritma Levenshtein Distance untuk Meningkatkan Efektivitas Pencarian Kata di Kamus Besar Bahasa Indonesia," *Teknik Elektro Journal* ,Vol. 7 No. 1, pp. 1-7, 2015.

[17] M. Ehmer Khan, "Different Approaches To Black box Testing Technique For Finding Errors", *International Journal of Software Engineering & Applications,* Vol. 2(4), pp. 31-40, 2016

[18] M. Tavakol and R. Dennick, "Making sense of Cronbach's alpha", *International Journal of Medical Education*. Vol. 2, pp. 53-5, 2011

[19] K. S. Taber, "The Use of Cronbach's Alpha When Developing and Reporting Research Instruments in Science Education", *Research in Science Education*, Springer, pp. 1-24, 2017.

[20] M. J. Warrens, "Five Ways to Look at Cohen's Kappa", *Psychology & Psychotherapy Journal,* Vol. 5 (4), 2015, pp. 1-4

[21] M. L. McHugh, "Interrater reliability: The kappa statistic", *Biochemia Medica Journal,* Vol. 22 (3), 2012, pp. 276-282