JACKA
Journal of Advanced Computer Knowledge and Algorithms
Vol. 2, No. 2, April, 2025, pp. 45-50
Journal homepage: http://ojs.unimal.ac.id/jacka

# Devayan Language Translator Dictionary Application Using the Levenshtein Distance Method on Android

Puja Faturrahman[1*], Zalfie Ardian[2], Maryana[3]

[1,2,3] Universitas Malikussaleh, Indonesia

[*]Corresponding Author Email: pujafaturrahman35@gmail.com

**Received: 19 December 2024**
**Revised: 20 March 2025**
**Accepted: 31 March 2025**
**Available online: 1 April 2025**

***Keywords:***
*Dictionary, Devayan Language, Levenshtein Distance, Android*

**ABSTRACT**

Indonesia is a country with a rich diversity of regional languages. Language is the primary tool of human communication, serving as a means to establish social relationships in daily life and as a medium for conveying information. One of the regional languages in Indonesia is Devayan, the native language of the people in Simeulue Regency. This language is used as a daily communication medium by the local residents. As one of Indonesia's archipelagic regions, Simeulue Island has various tourism potentials that attract tourists, workers, and students from outside the region. However, communication barriers often occur when visitors to Simeulue Island face difficulties interacting with the local community. Additionally, the language is gradually fading with the passage of time, as many young generations on Simeulue Island now have limited understanding of their regional language. Therefore, a dictionary application is needed to translate vocabulary from Devayan to Indonesian and vice versa. The Levenshtein Distance method is applied to the application's search feature, which has proven capable of correcting errors in input words and suggesting the closest words to users with an accuracy rate of 80.95%.

## 1. INTRODUCTION

Language serves as the primary medium of communication. Understanding language is crucial, as every conversation or interaction requires its use. Indonesia is the second country in the world with the highest number of languages, totaling 715. However, since Indonesian was established as the national language, there has been a shift in language usage that threatens the survival of regional languages. Currently, 440 languages in Indonesia are categorized as endangered [1].

Indonesia is an archipelagic country rich in ethnic, cultural, and linguistic diversity. Each language functions as a means of communication to support various forms of social interaction. Regional languages in Indonesia include Javanese, Acehnese, Sundanese, Madurese, Minangkabau, Betawi, Bugis, and others. Each province has its own diversity of regional languages, including Aceh Province, where many regional languages are spoken in different districts, such as the Devayan language. Devayan is a language spoken by an ethnic group living on Simeulue Island and is the native language of Simeulue Regency. It has been used as a means of communication by the local population since ancient times. Simeulue Regency is one of the archipelagic regions located 150 kilometers off the western coast of Aceh. The regency has diverse tourism potential, attracting tourists, workers, and students from outside the area. However, communication barriers often arise when visitors to Simeulue Island struggle to interact with the local community. Additionally, the language is gradually fading over time, as many young people on Simeulue Island have a limited understanding of their regional language. To preserve the language and provide a learning medium that supports understanding of regional languages, particularly Devayan, a tool capable of translating vocabulary, such as a dictionary, is needed. A dictionary serves as a medium to facilitate the search for word meanings or translations between languages [2]. With this application, it is expected to serve as a solution that facilitates interaction, so that language differences will no longer be a barrier to communication.

This research utilizes the Levenshtein Distance algorithm, which offers a higher level of accuracy and the ability to process data quickly. Its function is to compare identified words with input words and calculate the difference between them. The Android-based application is designed to provide benefits to the community across various categories, such as games, education, entertainment, and more. This has led to Android smartphones being widely used by different groups, particularly children and teenagers today.

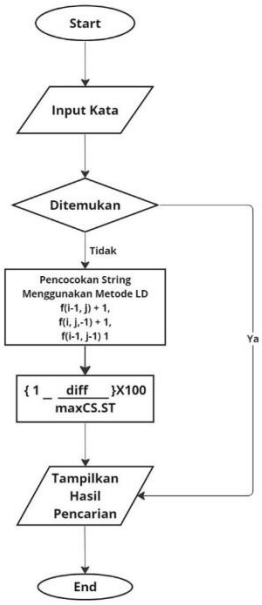
## 2. RESEARCH METHOD

### 2.1 Data Collection

This research was conducted in Simeulue Regency, Aceh Province. Data collection involved direct interviews with informants from the Simeulue Regency Tourism Office, responsible for culture and language, to obtain the necessary vocabulary in the Devayan language for the study. A literature review was conducted by gathering information related to the research, both offline and online, through reference books

about the Devayan language, as well as reading materials on language translation using the Levenshtein Distance method and its application in programming. The literature review included journals, books, and other reliable sources relevant to the research.

### 2.2 System Workflow




Gambar 2.1 Skema Sistem

The flowchart explains the following steps:

1. Input
   The user enters a word to search for, either for translation from Indonesian to Devayan or vice versa.
2. Database Check
   The system checks the database to verify whether the word entered by the user is available.
3. Translation or Suggestion:
   a. If the word is found, the system displays the corresponding word in the database along with its translation.
   b. If the word is not found, the system calculates the distance using the Levenshtein Distance method and suggests the closest word to the user.
4. Similarity Measurement
   The system calculates the similarity percentage between strings to determine the closeness of the match.
5. Result Display
   The search result is displayed, showing the entered word along with its translation in either Indonesian or Devayan.

### 2.3 Application

The term "application" originates from the English word "application," which means implementation, submission, or usage. Terminologically, an application refers to a program that is ready for use and designed to perform specific functions for users. It also serves to support other applications that can be used to achieve certain goals. According to the Executive Computer Dictionary, an application is defined as a solution to a problem that employs specific data processing techniques, generally focusing on desired computations or expected data processing [3].

### 2.4 Database

A database is a collection of data that is interconnected and related to one another. Database design refers to the process of planning for the creation and storage of data within a system, which consists of multiple database files [4].

### 2.5 Levenshtein Distance

The Levenshtein Distance algorithm is a method used to determine the minimum number of edit operations required to transform one string into another. This algorithm is commonly employed in approximate string matching. Introduced by Vladimir Levenshtein in 1965, the method calculates edit distance using a matrix to identify the number of differences between two strings. Levenshtein Distance is defined as the minimum number of operations, including insertion, deletion, and substitution of characters, needed to convert string A into string B. The distance between two strings is calculated based on the minimum number of these operations required [5].

The Edit Distance algorithm is calculated using a matrix that serves to determine the number of differences between two strings. As an example of the application of this algorithm, the strings "blajar" and "belajar" have a distance of 1 because only one operation is required to transform one string into the other. In this case, the string "blajar" can be changed to "belajar" by inserting the character "e" at the second position of the source string [6]. Here are the 3 operations in the Edit Distance algorithm:

1. Character Substitution

This operation involves replacing one character with another. Below is a substitution matrix table for character replacement:

Table 2.1 Character Substitution

| String 1 | b | e | l | a | j | a | a |
|---|---|---|---|---|---|---|---|
| String 2 | b | e | l | a | j | a | r |

In the table above, the word "belajaa" indicates a typing error where the character "a" at the sixth position of string 1 should be replaced with the character "r". In this case, the substitution operation is needed to replace the incorrect character with the correct one. Using the substitution operation, the character "a" at the sixth position in "belajaa" is substituted with "r", transforming the string into "belajar". This operation ensures that the strings are the same, and the Edit Distance algorithm counts this as one edit.

2. Character Insertion

This operation involves inserting a character into a string. Below is the insertion matrix for character insertion:

Table 2.2 Character Insertion

| String 1 | b | e | - | a | j | a | r |
|---|---|---|---|---|---|---|---|
| String 2 | b | e | l | a | j | a | r |

In the table above, the word "beaajar" in the first string is not fully typed because it is missing one character, which is "l". In this case, the insertion operation is required to add the missing character "l" to the string. This will make the first string match the second string, which is "belajar". The

insertion operation accounts for the missing "l", making the Edit Distance between the two strings 1.

3. Character Deletion

This operation involves removing a character from a string. Below is the deletion matrix for character removal:

Table 2.3 Character Deletion

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| String 1 | b | e | l | a | j | a | r | h |
| String 2 | b | e | l | a | j | a | r | - |

In the table above, the word "belajarh" in the first string has an extra character "h" at the end, which is not present in the second string, "belajar". In this case, the deletion operation is required to remove the extra character "h" to make the first string match the second one. The deletion operation accounts for removing the "h" character, making the Edit Distance between the two strings 1.

The process of calculating the distance using the Levenshtein Distance algorithm begins from the top-left corner of a two-dimensional matrix, which contains the characters of the source string and the target string. Each element in the matrix is assigned a cost value. These values reflect the number of deletion, insertion, and substitution operations required to transform the source string into the target string. Operations such as deletion, insertion, substitution, and no change in characters can be explained through Equations (1) to (4) [7].

$$lev(a,b) = \min lev\ (a-1,b) + 1 \quad \textbf{(1)}$$
$$lev(a,b) = \min lev\ (a,b-1) + 1 \quad \textbf{(2)}$$
$$lev(a,b) = \min lev\ (a-1,b-1) + 1, aj \neq bi \quad \textbf{(3)}$$
$$lev(a,b) = \min lev\ (a-1,b-1) + 1, aj = bi \quad \textbf{(4)}$$

The process and calculation between strings in the equation can be seen in the following matrix table:

Table 2.4 Levenshtein Distance Calculation

| | t | K | O | M | P | R | E | H | E | N | S | I | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| s | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| I | 1 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 10 | 11 |
| N | 2 | 2 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 8 | 9 | 10 | 11 |
| T | 3 | 3 | 3 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| E | 4 | 4 | 4 | 4 | 4 | 5 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| N | 5 | 5 | 5 | 5 | 5 | 5 | 6 | 7 | 7 | 7 | 8 | 9 | 10 |
| S | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 7 | 8 | 8 | 7 | 8 | 9 |
| I | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 8 | 9 | 10 | 7 | 8 |
| F | 8 | 8 | 8 | 8 | 8 | 8 | 8 | 8 | 8 | 9 | 10 | 8 | 7 |

The table above works by comparing two strings and adding a value of 1 for each character that differs between the two strings, where (s) refers to the source string and (t) refers to the target string. The operations performed in Table 2.4 can be seen in the following table.

Table 2.5 Levenshtein Distance Operations

| (s) | (t) | Insertion | Deletion | Substitution | Distance |
|---|---|---|---|---|---|
| INTENSIF | KOMPREHENSIF | P, R, E, H | | I > K<br>N > O<br>T > M | 7 |

After the Levenshtein distance between the two compared words is successfully calculated, the similarity level of the comparison between the two words is determined using the following equation:

$$Sim = 1\frac{d[m,n]}{\max[s,t]}\ x\ 100\%$$

Note:

*Sim* = *Similarity*
*d [m, n]* = *Levenshtein Distance*
*max [s, t]* = Maximum Number of Character from 2 Strings

## 3. RESULT AND DISCUSSION

### 3.1 Research Results

The Levenshtein Distance method is applied to the vocabulary search menu. When the user presses the available Search button, the system will calculate the distance between each word in the database and the searched word. Then, the system will display recommendation results based on the string distances calculated using the Levenshtein Distance method. The vocabulary with the smallest Levenshtein distance will be selected as the suggested word, such as "kelapa" for the search "klapa," which has a distance of 1. Below is a table showing the Levenshtein distance for Indonesian words and the operations and calculations for each word.

Table 3.1 Indonesian Word Distance

| Input | Recommendation | Levenshtein Distance |
|---|---|---|
| klapa | kelapa | 1 |
| | apa | 2 |
| | kapan | 2 |
| | kaya | 2 |
| | kenapa | 2 |
| | lama | 2 |
| | lapar | 2 |
| | siapa | 2 |

Table 3.2 Operations Details on the Recommended Word

| Input | Recommendation | Insertion | Deletion | Substitution | Distance |
|---|---|---|---|---|---|
| klapa | kelapa | a | | | 1 |
| | apa | | k, l | | 2 |
| | kapan | n | l | | 2 |
| | kaya | | l | p>y | 2 |
| | kenapa | n | | l>e | 2 |
| | lama | | k | p>m | 2 |
| | lapar | r | k | | 2 |
| | siapa | | | k>s<br>l>i | 2 |

Table 3.3 Word Search Similarity Results

| Input | Recommendation | D [m, n] | Max [S, T] | Similarity |
|---|---|---|---|---|
| klapa | kelapa | 1 | 6 | 83,33% |
| | apa | 2 | 6 | 66,67% |
| | kapan | 2 | 6 | 66,67% |
| | kaya | 2 | 6 | 66,67% |

| | | | | |
|---|---|---|---|---|
| | kenapa | 2 | 6 | 66,67% |
| | lama | 2 | 6 | 66,67% |
| | lapar | 2 | 6 | 66,67% |

After all the Levenshtein distance values are calculated, it can be concluded that the string with the highest similarity value is obtained from the lowest distance value. In this example, the vocabulary with the lowest distance value and the highest similarity value to be recommended is the word "kelapa".

### 3.2 Accuracy Evaluation Results

The Levenshtein Distance algorithm is applied for vocabulary search and error correction on the input words. The application is evaluated based on the number of correctly recommended words from 50 test cases using different scenarios. A recommendation is considered correct (Y) if it matches the intended vocabulary by the user, and considered incorrect (N) if it does not match. The accuracy of the algorithm is evaluated manually by repeating the search for 50 pairs of Devayan language vocabulary words that were intentionally written randomly to be used as test data. Meanwhile, the original data is used as accurate data. The results of the accuracy evaluation can be seen in Table 3.4.

Table 3.4 Accuracy Evaluation Results

| No | Scenario | Input | Meaning | Recom | Note |
|---|---|---|---|---|---|
| 1 | Substitute 1 char | afatek | afitek | afitek | Y |
| 2 | Delete 1 char | afort | aforat | aforat | Y |
| 3 | Substitute 1 char | ancifa lul | ancifalal | ancifalal | Y |
| 4 | Delete 1 char | ancolf i | ancolafi | ancolafi | Y |
| 5 | Delete 1 char | anku | angku | angku | Y |
| 6 | Delete 1 char | ante | antek | antek | Y |
| 7 | Delete 1 char | bhak | bahak | bahak | Y |
| 8 | Delete 1 char | badu | bakdu | bakdu | Y |
| 9 | Substitute 1 char | bilal | balal | balal | Y |
| 10 | Substitute 1 char | bangk ang | bangken g | bangken g | Y |
| 11 | Insert 1 char | emhen | ehen | ehen | Y |
| 12 | Insert 2 char | maena k | enak | anak | N |
| 13 | Substitute 2 char | engal | enggel | enak | N |
| 14 | Delete 3 char | enkan o | engkanen g | Not Found | N |
| 15 | Insert 2 char | ahhik k | ahik | ahik | Y |
| 16 | Substitute 1 char | batu | fatu | fatu | Y |
| 17 | Delete 1 char | fegi | fengi | fengi | Y |
| 18 | Substitute 2 char | bsang | fesang | asang | N |
| 19 | Substitute 1 char | bisil | fisil | fisil | Y |
| 20 | Delete 2 char | haban g | halubang | sabang | N |
| 21 | Substitute 1 char | enom | inom | inom | Y |
| 22 | Insert 2 char | inting e | iting | iting | Y |
| 23 | Delete 1 char | khan | kahan | kahan | Y |
| 24 | Delete 1 char | keldek | kaledek | kaledek | Y |
| 25 | Substitute 1 char | haol | kaol | haok | N |
| 26 | Delete an 1 char | arano | karano | karano | Y |
| 27 | Delete an 1 char | ebau | kebau | kebau | Y |
| 28 | Delete an 1 char | uak | kuak | kuak | Y |
| 29 | Delete 2 char | laleng | lahaleng | kabeng | N |
| 30 | Insert 1 char | lalatu n | lalatu | lalatu | Y |
| 31 | Delete 1 char | enai | nai | enak | N |
| 32 | Delete 2 char | nciuk | nanciuk | nciuk | Y |
| 33 | Delete 1 char | nrako | narako | narako | Y |
| 34 | Insert 1 char | nauu | nau | nau | Y |
| 35 | Substitute 2 char | nenah o | nemafo | nemafo | Y |
| 36 | Insert 1 char | nngan g | ngang | ngang | Y |
| 37 | Substitute 1 char | ueng | oeng | oeng | Y |
| 38 | Insert 1 char | ofhel | ofel | ofel | Y |
| 39 | Insert 2 char | looko k | okok | okok | Y |
| 40 | Insert 1 char | ollol | olol | olol | Y |
| 41 | Substitute 1 char | rumair | rumaar | rumaar | Y |
| 42 | Substitute 1 char | lumek | rumek | rumek | Y |
| 43 | Delete 1 char | sahli | sahuli | sahuli | Y |
| 44 | Substitute 2 char | sekeja b | sakajab | sakajab | Y |
| 45 | Substitute 2 char | selada n | salapan | salapan | Y |
| 46 | Delete 2 char | saluncg | saluncun g | saluncun g | Y |
| 47 | Substitute 2 char | safilan | sambilan | sambilan | Y |
| 48 | Substitute 1 char | cangki r | sangkir | sangkir | Y |

| | | | | | |
|---|---|---|---|---|---|
| 49 | Delete 1 char | slafai | silafai | silafai | Y |
| 50 | Insert 1 char | silail | silai | silai | Y |

Based on the table above, it can be seen that the Input column contains the words entered by the user, the Intended column shows the words desired by the user, and the Recommendation column displays the word recommendations provided by the system. Out of 50 test data, there were 42 words that were correct (Y) and 8 words that were incorrect (N). This is due to some target strings having a Levenshtein Distance that is too far, or some target strings having the same similarity value. Below is the accuracy level of the Levenshtein Distance algorithm:

$$accuracy = \left(\frac{42-8}{42}\right) \text{x } 100\% \quad = 80{,}95\%$$

After evaluating the accuracy of 50 data points, it was found that the Levenshtein Distance algorithm has an accuracy rate of 80.95% in correcting spelling errors and providing the closest word recommendations to users.

### 3.3 Application Interface

1. Main Activity

This page contains several available menus, including translations from Indonesian to Devayan and from Devayan to Indonesian.


Figure 1. Main Activity

2. Indonesia – Devayan Page

This page displays the Indonesian - Devayan translation. Users can search for and view a list of Indonesian words and their Devayan translations by clicking on the target word.

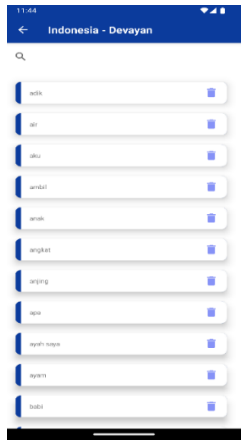



Figure 2. Indonesia – Devayan Page

3. Devayan – Indonesia Page

This page displays the Devayan - Indonesian Search. Users can search for and view a list of Devayan words and their Indonesian translations by clicking on the target word.

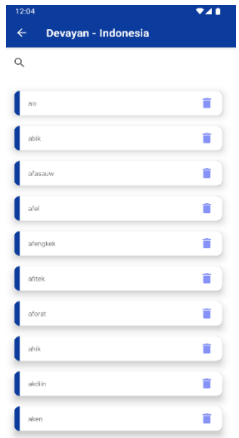



Figure 3. Devayan – Indonesia Page

4. Translation Results Page

After the user clicks on the desired word, the translation results from Indonesian to Devayan will appear on the screen.

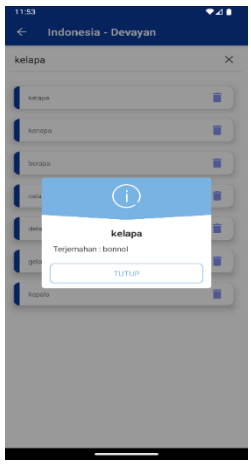



Figure 4. Translation Result Page

5. Add Data Page

On this page, users can add new vocabulary along with its translation into the application.


Figure 5. Add Data Page

## 4. CONCLUSION

Levenshtein Distance in the word search feature of the Devayan language translator dictionary application for Android can improve the performance of word searches, making them more accurate and efficient. The data used in this study consists of 500 words that cover two languages: Indonesian and Devayan. Some key conclusions from this research are as follows:

1. The developed dictionary application can be effectively used to translate vocabulary between Devayan and Indonesian, which can facilitate communication between the Simeulue community, who have limited knowledge of Indonesian, and newcomers who are unfamiliar with the Devayan language.
2. The Levenshtein Distance method applied can correct spelling errors in the input words by calculating the distance between words, and provide the closest word recommendations to be displayed to users.
3. The accuracy level of the Levenshtein Distance method in correcting the intended words by the user is quite good, with an accuracy rate of 80.95%.

## REFERENCES

[1] I. A. Laia, "Rancang Bangun Aplikasi Penerjemah Bahasa Indonesia Bahasa Nias Menggunakan Algoritma Levenshtein Distance Berbasis Android," 2023.

[2] A. Fitriya, "Pengaruh Media Pembelajaran Kamus Online Al-Ma'āny Dalam Meningkatkan Mahārah Al-Qirā'ah Pada Siswa Kelas X Man 2 Sleman Tahun Ajaran 2017/2018 (Studi Eksperimen)," vol. 5, no. 1, 2020.

[3] D. Abdullah, Kahirulasmi, and I. G. A. K. Warmayana, Perancangan & Pembuatan Aplikasi File Server Berbasis Web Menggunakan Metode Interpolation Search. 2020.

[4] H. Zulfikar and Z. Yunizar, "Sistem Informasi E-Commerce Berbasis Web Dan Android Study Kasus Toko Jakarta Elektronik Samalanga," Apr. 2021.

[5] R. D. Priyatna, "Implementasi Algoritma Levenstein Distance Dan Algoritma Knutt Morris Pratt Dalam Fitur Word Completion Pada Search Engine," 2023.

[6] A. Rofiqih, M. Z. Siambaton, and T. Haramaini, "Penerapan Algoritma Levenshtein String pada E-Arsip Kecamatan Pagar Merbau," *sudo Jurnal Teknik Informatika*, vol. 1, no. 1, pp. 1–7, Mar. 2022, doi: 10.56211/sudo.v1i1.1.

[7] A. Ardiansyah, "Implementasi Algoritma Levenshtein Distance pada Aplikasi Kamus Terjemahan Bahasa Indonesia-Bahasa Bangka," 2023.