INTENSIF, Vol.7 No.2 August 2023
**ISSN: 2580-409X (Print) / 2549-6824 (Online)**
**DOI:** https://doi.org/10.29407/intensif.v7i2.20103

# Comparison of Modified Nazief&Adriani and Modified Enhanced Confix Stripping algorithms for Madurese Language Stemming

**Received:**
21 May 2023
**Accepted:**
25 July 2023
**Published:**
1 August 2023

**[1*]Enni Lindrawati, [2]Ema Utami, [3]Ainul Yaqin**
*[1,2]Magister of Informatics Engineering, Universitas Amikom Yogyakarta*
*[3]Faculty of Computer Science, Universitas Amikom Yogyakarta*
*E-mail: [1]ennilindra@students.amikom.ac.id, [2]ema.u@amikom.ac.id, [3]ainulyaqin@amikom.ac.id*

*Corresponding Author

**Abstract—**The Madurese language has a unique morphology. The morphological uniqueness can be used to find basic words. The basic word process is called stemming. Stemming can be developed into an application for translating Madurese into Indonesian and even other languages. It can support the development of a Madurese language text plagiarism system. Stemming research on the Madurese language is still rare. Therefore, this study aims to find the basic words of the Madurese language using modifications to the Nazief & Adriani algorithm and Enhanced Confix Stripping (ECS) modifications. The study used 1000 Madurese words, consisting of 630 prefix words, 74 ending words, and 296 confix words. The results showed that the modification of the Nazief & Adriani algorithm was better, shown by the accuracy obtained of 88.8% with overstemming of 0.7% and understemming of 10.5%. As for ECS, an accuracy of 74.0% was obtained, 0.4% overstemming, and 25.6% understemming. In the same process, Nazief&Adriani's modification is faster than the ECS modification. For the Nazief&Adriani modification, it takes 13.31 seconds while for the ECS modification, it takes 210.88.
**Keywords**— Madurese; Morphology; Stemming; Nazief & Adriani; ECS

This is an open access article under the CC BY-SA License.

***Corresponding Author:***
Enni Lindrawati
Magister of Informatics Engineering
Universitas Amikom Yogyakarta
Email: ennilindra@students.amikom.ac.id
ID Orcid: http://orcid.org/0009-0000-0684-0228

**ISSN: 2580-409X (Print) / 2549-6824 (Online)**
**DOI:** https://doi.org/10.29407/intensif.v7i2.20103

# I. INTRODUCTION

Indonesia is a country consisting of various tribes, races, languages, religions and different cultures [1]. This diversity is God's gift that is not owned by other countries. Language diversity cannot be separated from culture because language is part of the culture [2], including the Madurese language. The Madurese language is known as a regional or local language. The Madurese language is no stranger to Indonesian people because this language is widely spoken on the island of Java, for example, in Banyuwangi, Probolinggo, Jember, Bondowoso and Situbondo [3].

The Madurese language has much uniqueness, especially in the formation of its morphology. Morphology is a scientific discipline that studies language units as grammatical units. The process of morphology is the process of forming words that emerge from their basic forms through affixes, repetitions, and compounding [4]. Various kinds of affixes ( *embuwen)* in Madurese, namely *ter-ater* (prefix), *panotèng* (suffix), *oca' pangadâ' ècampor panotèng* (confix) [5]. In the formation of basic words, there are changes in sound or phonemes. This process is called morphophonemic [6].

These basic lexical rules can be used to build an Information Retrieval application, and you will surely encounter problems implementing it. Information Retrieval is a material (document) search method, including the process of Searching for and retrieving knowledge-based information from a collection of documents [7] as well as methods for presenting, storing, and retrieving large data sets for data mining and access to find relevant results that meet user needs in response to user requests [8].

This problem requires an application that finds the roots of words in Madurese by stringing together ter-ater (prefix), panotèng (ending), oca' pangadâ' ècampor panotèng (confix). With the development of technology and science, it is advisable to use text mining with the stemming method to determine the basic words of the Madurese language. According to Feldman (2007), text mining is a data-intensive process in which users interact and work with documents using various analytical tools, while derivation is a process in which word forms are mapped and removed so that the word belongs to it [9 ].

Stemming is a basic word search technique by removing the affixes in the word [10]. The problems that often occur in the stemming process are overstemming and understemming. Overstemming is a process of cutting off excess affixes in a word resulting in words with different meanings [11]. In contrast, understemming does not occur with the cutting of affixes, so the resulting word is the same as the original word, which still has affixes [12].

**ISSN: 2580-409X (Print) / 2549-6824 (Online)**
**DOI:** https://doi.org/10.29407/intensif.v7i2.20103

Previous research on language stemming is still minimal, which has been done by [13] and [14]. Both studies used a modified Enhanced Confix Stripping (ECS) algorithm. The second research shows that the ECS performance is quite good, with an accuracy of 95.75% with 400 trial words. At the same time, the first study did not include the results of accuracy. The Madurese language stemming research using a modification of the Nazief & Adriani algorithm has never been done. However, modifications of the Nazief & Adriani algorithm are often carried out in research on the Java language.

Several studies have explored the application of the Nazief & Adriani algorithm in analyzing the Javanese language [15] [16] [17] [18]. Similarly, the algorithm has been used for ECS (Exact Compound Splitting) [19]. Additionally, researchers have also investigated the application of the Nazief & Adriani algorithm in languages other than regional languages [20] [21] [22] [23]. Notably, the Madurese language, which belongs to the West Austronesian language family, exhibits significant phonological and lexical similarities to Javanese [24]..

Based on this description, the researcher will make a comparison of the modified Nazief & Adriani algorithm and the ECS modification in Madurese by not removing the accents in words so that the phonology of words does not change the meaning of words according to the latest spelling of Madurese [25] a basic word search application can be developed Madurese.

## II. RESEARCH METHOD

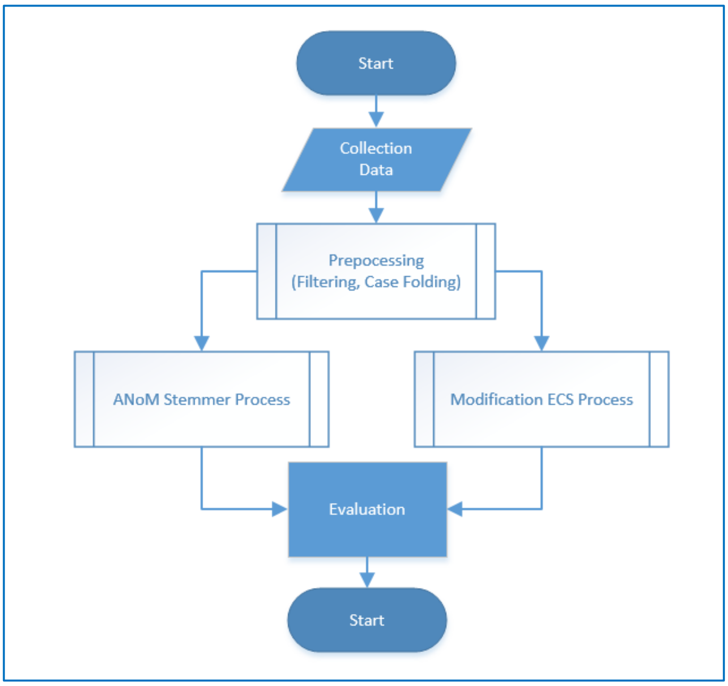


**Fig 1.** Research System Flowchart

**ISSN: 2580-409X (Print) / 2549-6824 (Online)**
**DOI:** https://doi.org/10.29407/intensif.v7i2.20103

**A. Data collection**

The data used in this study are words with Madurese affixes of 1000 and 2259 basic words taken in the Madurese language dictionary from contemporary Madurese-Indonesian Dictionary by Muhri, S.Pd, M.A.

**B. Preprocessing**

This stage makes the text structured [26] so that the data is ready to be processed [27], the steps are as follows:

1. Filtering.

   Filtering changes accented letters to regular letters of the alphabet.
2. Case Folding.

   This process converts all text letters to all lowercase [28].

**C. Process Modification Stemming Nazief & Adriani (ANoM Stemmer)**

Three main steps Nazief & Adriani [29] :

1. Removing affixes (affixations) following the morphological rules of the Indonesian language.
   a. Inflectional Endings:
      - Pronouns _ possessive (PP) {"-me", "-mu", "-nya"}

   b. Derivation suffixes (DS) {"-i", "-kan", "-an"}

   c. Derivation prefixes (DP) :
      - DP morphology : {"me-", "be-", " te -", "pe-"}
      - Normal DP: {"at-", " to -", "se-"}
2. The algorithm is highly dependent on the basic word dictionary
3. Supports recording by returning overstemmed words.

There are two modifications in the stemming process:

1. Eliminates the process of checking particles because writing particles in Madurese is not attached to other words.
2. Eliminates unauthorized affixes.

Based on the Nazief & Adriani principles, the following is ANoM Stemmer method for the Madurese language:

1. Removing affixes (affixations) according to the morphological rules of the Madurese language
   a. Inflectional Endings:
      - Pronouns _ possessive (PP) {"-na"}

INTENSIF, Vol.7 No.2 August 2023
**ISSN: 2580-409X (Print) / 2549-6824 (Online)**
**DOI:** https://doi.org/10.29407/intensif.v7i2.20103

b. Derivation suffixes (DS) {"-a," "-e," "-na," "-ana," "-an," "-aghi,"}

c. Derivation prefixes (DP) :

- DP morphology: {" N- " } which is called anuswara affix, namely {"nya-," "ma-," "nga-," "-na,"}
- Common derivation prefixes: {" a -", " ta -", “ma-” “ka-”, “sa-”, “pa-”, “pe-”, “pre-”, “nga-”, “ -e", "-epa", "-eka"; “”}.

2. The algorithm is highly dependent on the basic word dictionary
3. Supports recording by returning overstemmed words.

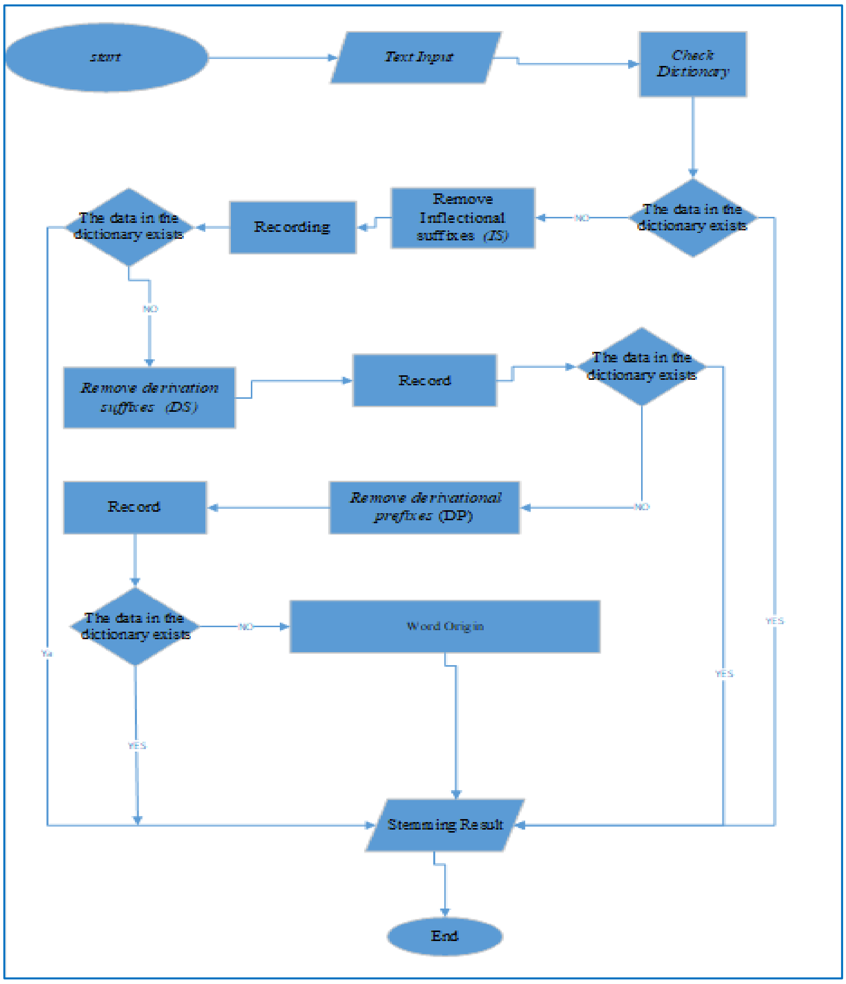


**Fig 2.** ANoM Stemmer Process Flowchart

INTENSIF, Vol.7 No.2 August 2023
**ISSN: 2580-409X (Print) / 2549-6824 (Online)**
**DOI:** https://doi.org/10.29407/intensif.v7i2.20103

## D. Process Enhanced Confix Stripping (ECS)

Enhanced Confix Stripping (ECS) algorithm steps [ 26] :

1. Enter a word.
2. The dictionary checks the word entered. If the entered word is found in the dictionary, it will be considered the base word. If not, the process of removing the last word is performed.
3. The third process is removing the last word from the input word. If the third process results in a word in the dictionary, the word is considered a root word. Otherwise, the prefix removal process is complete.
4. The fourth process is prefix removal. If the result of the fourth process is a word found in the dictionary, it is considered a root word. However, if the word is not found in the dictionary, recoding is performed.
5. If the word transcoding process is found in the dictionary, the process stops.
6. The registration process is the act of breaking down or rearranging words that have undergone additional derivation processes.
7. If it still fails, continue with the last recovery or recovery.
8. If it fails, the word is returned to its original form and considered a root word.

There are two modifications in the stemming process:

3. Eliminates the process of checking particles because writing particles in Madurese is not attached to other words.
4. Eliminates unauthorized affixes.

Steps to modify the Enhanced Confix Stripping (ECS) algorithm for the Madurese language:

1. Enter a word.
2. Check the dictionary. If the word is in the dictionary, the root word has been found.
3. Remove the inflected suffix (particle) { "la," " kan," "ya," " jha,'", “ra” }
4. Omit possessive pronouns {-na}
5. Check dictionary
6. Eliminating prefixes {" a- ", " ta ", “ma-” “ka-”, “sa-”, “pa-”, “pe-”, “pre-”, “nga-”, “e-”, “epa-”, “eka-”}
7. Check the dictionary
8. Omit derivation endings (suffix) {"-a," "-e," "-na," "-ana," "-an," "-aghi,"}
9. If it still fails, continue with the last recovery or recovery.
10. If it fails, the word is returned to its original form and considered a root word.

INTENSIF, Vol.7 No.2 August 2023
**ISSN: 2580-409X (Print) / 2549-6824 (Online)**
**DOI:** https://doi.org/10.29407/intensif.v7i2.20103

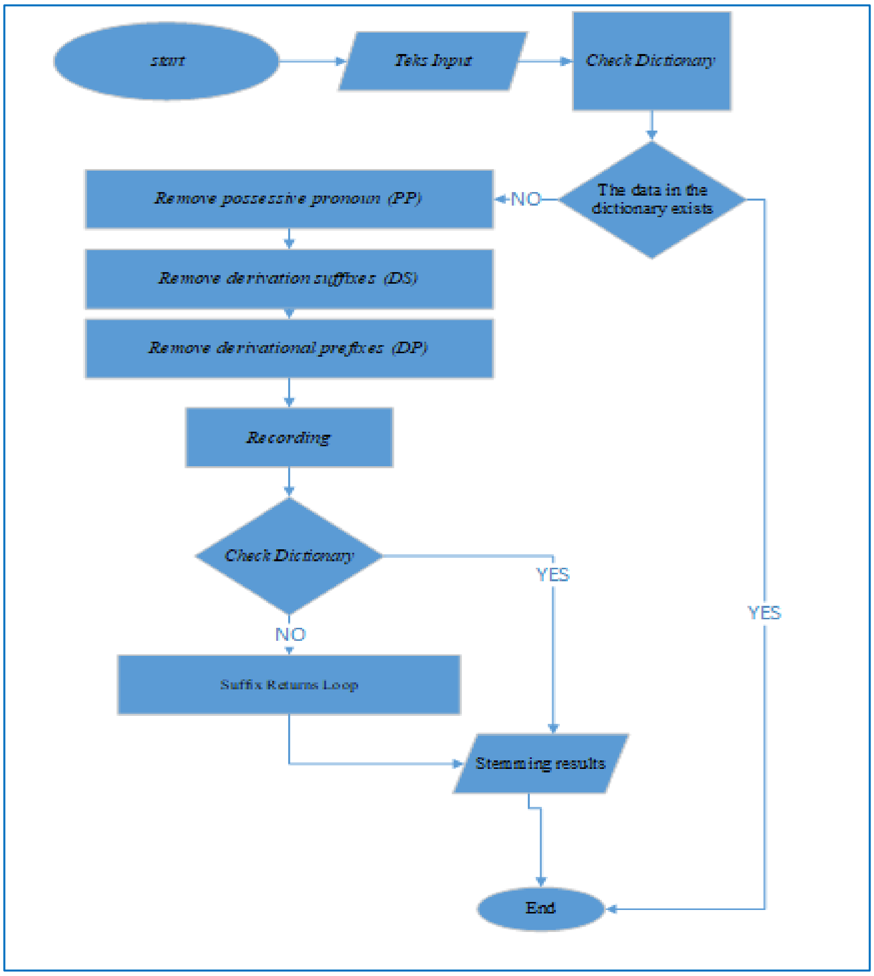


**Fig 3.** ECS Modification Process Flowchart

The rules for beheading affixed words in the Madurese language is presented in table 1[14] :

**Table 1.** Rules of Beheading of Madurese Affixed Words

| No | Word Formats | Word Fragment | Example words |
|---|---|---|---|
| 1 | eA | e-A | eatorè, ecokor, etapok |
| 2 | aA | a-A | aghâbây, alonca‟, akemmor |
| 3 | taA | ta-A | tabâca, talèbât, tapalè‟ |
| 4 | maA | ma-A | mataber, maalos, mapotè |
| 5 | kaA | ka-A | kaator, katello, kasebbhut |
| 6 | saA | sa-A | saalam, saarè, sakampong |
| 7 | paA | pa-A \| pa-nV | paloros, pakonèng, patao |
| … | … | … | … |
| 38 | CCepon | C-Cepon | pèccottepon, |

INTENSIF, Vol.7 No.2 August 2023
**ISSN: 2580-409X (Print) / 2549-6824 (Online)**
**DOI:** https://doi.org/10.29407/intensif.v7i2.20103

In Table 1, Cand A is the consonant. A can also be a vowel just like V. Lastly, P is a fragment of a word.

**E. Evaluation**

Evaluation is calculated by dividing the number of words stemming correctly by the total number of words multiplied by 100% [30] .

## III. RESULT AND DISCUSSION

**A. Data collection**

This research uses Python to develop the model. The input for this model is the word Madura added with affixes. There are 1000 words with 630 prefixes, 74 suffixes and 296 affixes. The percentage of prefixes, suffixes and confixes can be seen in figure 2 below from Contemporary Madurese-Indonesian Dictionary by Muhri, S.Pd,. M.A

**B. Preprocessing Results**

In Table 2 it can be seen the results of preprocessing.

**Table** 1. PREPROCESSING RESULTS

| No | Word | Results | Meaning |
|---|---|---|---|
| 1 | abhisan | abhisan | beg |
| 2 | abhubu | abhubu | donate |
| 3 | abhuctè | abhukte | proven |
| 4 | abhundhu ' | abhundhu ' | wrapped |
| 5 | sellibhan | sellibhan | machine skid |
| 6 | sèmpennan | simpennan | savings |
| 7 | siyong | siyong | i ka n catfish |
| 8 | adhuwaaghi | adhuwaaghi | pray |
| 9 | adaâtengngan | adatengan | arriving |
| ... | ... | ... | ... |
| 1000 | Tabhalighâ | Tabhalighha | vice versa |

**C. Stemming Results**

1. Prefix stemming results

At this stage, a test was carried out on the essential words of the Madura language prefix in 630 words. An example of the word is "akalambhi" so the stemming results obtained are "kalambhi" where the word comes from the prefix "a-" and the word "kalambhi". The following table 3 stemming results for prefix.

INTENSIF, Vol.7 No.2 August 2023
**ISSN: 2580-409X (Print) / 2549-6824 (Online)**
**DOI:** https://doi.org/10.29407/intensif.v7i2.20103

**Table** 3. Prefix Stemming Results

| No | Word | ANoM Stemmer | | ECS | |
|---|---|---|---|---|---|
| | | Results | Status | Results | Status |
| 1 | aata ' | ata ' | success | ata ' | success |
| 2 | abaddha | bâḍḍhâ | success | bâḍḍhâ | success |
| 3 | abâgi | bâgi | success | bâgi | success |
| 4 | abai ' | bai ' | success | bai ' | success |
| 5 | abâjâng | bajâng | success | bajâng | success |
| 6 | abâkto | bâkto | success | bâkto | success |
| 7 | abâlâ | bâlâ | success | abâlâ | Understemming |
| 8 | abâlânjha | bâlânjha | success | abâlânjha | Understemming |
| 9 | abângon | bângon | success | Bângon | success |
| 10 | abârna | abâr | Overstemming | bârna | success |
| 11 | abâtek | bâtek | success | bâtek | success |
| 12 | abhalik | abhalik | Understemming | abhalik | Understemming |
| .... | .... | ..... | .... | .... | ..... |
| 630 | tasoddhu ' | soddhu ' | success | soddhu ' | success |

From Table 3, there are several errors, namely the word "abârna" in the Nazief & Adriani modification occurs overstemming because it is read as having the ending "-na," so the resulting word is "abâr." Whereas the ECS modification produces the correct word, namely "bârna." Furthermore, in the word "abâlâ" for the successful modification of Nazief & Adriani, the word produced is "bâlâ" whereas, in the ECS modification, there is understemming because it fails to read the prefix "a-," while the word does not contain an ending so it is returned to the original base word.

2. Suffix stemming results

At this stage, a test was carried out on the basic words of the Madura language prefix in a total of 74 words. An example of the word is "labângnga," so the stemming results obtained are "labâng," where the word comes from the word "labâng" and the ending "-an." In the process of its formation, if the ending "-an" meets a root word that ends in "ng," then the formation of the word will form "ngng." The following table 4 stemming results for suffix.

**Tablel 4.** Suffix Stemming Results

| No | word | ANoM Stemmer | | ECS | |
|---|---|---|---|---|---|
| | | Results | Status | Results | Status |
| 1 | abbhana | abbhana | Understemming | abbhana | understemming |
| 2 | abdhina | abdhi | success | abdhina | understemming |
| 3 | abidhana | abidhana | Understemming | abidhana | understemming |
| 4 | addhuân | addhu | success | addhuân | understemming |
| 5 | afalan | afal | success | afalan | understemming |
| 6 | bângalan | bângal | success | bângal | success |
| 7 | battonna | battonna | Understemming | battonna | understemming |
| .... | .... | .... | .... | .... | .... |
| 74 | masana | mas | Overstemming | pas | overstemming |

**ISSN: 2580-409X (Print) / 2549-6824 (Online)**
**DOI:** https://doi.org/10.29407/intensif.v7i2.20103

In Table 4, a lot of understemming occurs in the ECS modification because the retrieval of the "-an" ending in the base word fails, so it is returned to the original base word. For the word "massana," in both methods, there was an error, namely understemming, because in the modification Nazief & Adriani read, there was an ending "-ana". Meanwhile, the ECS modification assumes the prefix "N-" and the suffix "-ana."

3. Confix Stemming results

At this stage, a test was carried out on the basic words of the Madurese language konfix totaling 296 words. An example of the word is "kajhujhurân" so the stemming results obtained are "jhujhur," where the word comes from the word "jhujhur," which gets the prefix "ka" and the ending "-an." The following table 5 stemming results for suffix.

**Tablel 5.** Suffix Stemming Results

| No | word | ANoM Stemmer | | ECS | |
|---|---|---|---|---|---|
| | | Results | Status | Results | Status |
| 1 | katorodhan | katorodhan | Understemming | katorodhan | understemming |
| 2 | kawâjibhan | kawâjibhan | Understemming | kawâjibhan | understemming |
| 3 | mabhakalè | bhakal | success | bhakal | success |
| 4 | mabinèè | bine | success | bin | success |
| 5 | eat | eat | Understemming | feed | success |
| 6 | malakèe | lakè | success | malakèe | understemming |
| 7 | mangabbherraghi | mangabbherraghi | Understemming | mangabbherraghi | understemming |
| 8 | nabaraghi | tabar | success | be patient | success |
| 9 | nako'è | tako ' | success | tako ' | success |
| 10 | naleè | tale | success | tale | success |
| .... | .... | .... | .... | .... | .... |
| 269 | Tabhalighâ | bhalik | success | Tabhalighâ | understemming |

In Table 5, it can be seen that understemming often occurs, in this case it shows that the recording process in the search loop for the root word is more on the initial base word, so the root word was originally considered a root word.

## D. Evaluation

Both of these algorithms are tested for their performance using accurate calculations with the following equation 1 [26] :

$$\text{Accuracy} = \frac{Correct\ Member\ of\ Word}{Total\ Member\ of\ Word} x100\% \quad (1)$$

**ISSN: 2580-409X (Print) / 2549-6824 (Online)**
**DOI:** https://doi.org/10.29407/intensif.v7i2.20103

The stemming trial results is presented in Table 6:

**Tablel 6.** Stemming Trial Results

| No | Affix | ANoM Stemmer | | | | ECS | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | ***Correct*** | ***Over stemming*** | ***Understemming*** | ***accuracy*** | ***Correct*** | ***Over stemming*** | ***Understemming*** | ***accuracy*** |
| 1 | Prefix | 589 | 4 | 37 | 93.49 % | 519 | 2 | 110 | 82.38 % |
| 2 | Suffix | 62 | 1 | 11 | 83.78 % | 44 | 1 | 29 | 59.46 % |
| 3 | Confix | 237 | 2 | 57 | 88.10 % | 177 | 1 | 118 | 65.80 % |
| 4 | All ( Prefix, Suffix ,Confix ) | 888 | 6 | 105 | 88.80 % | 740 | 4 | 257 | 74.00 % |

Table 6 shows the highest accuracy in the prefix test, the second highest in confix, and finally in the suffix. For testing on ECS it is smaller than previous studies [14], this study uses 400 basic words as root words. However, this study has been able to read anuswara prefixes or endings so that words that experience morphophonemic produce appropriate base words, reducing the unreadable smelting process such as "ng" to "k" in research [17] the word "Nithik" failed to become "thithik."

Even in the stemming process, ANoM Stemmer requires less time than the ECS modification process. For the Nazief&Adriani modification, it takes 31.31 seconds while for the ECS modification, it takes 210.88 seconds. This is because the ECS algorithm has a repeated decapitation process as long as it is still possible to decapitate the affixes, any excess stemming will be recorded, if the recording is not in the dictionary it will be returned to the original root word which will result in incorrect stemming results or understemming.

Algorithm modifications have also not been able to resolve infixes and reduplications. So it is necessary to add insertion and repetition rules according to the morphology of the Madurese language. Combining the two methods in research can also improve performance, as was done [17].

To increase the success of stemming in each method, it is necessary to collect more basic words according to the rules or those obtained from the Madurese language dictionary. As well as increasing the number of references to the rules for forming words in the Madurese language, so that the stemming rules for words that fail in the stemming process can be corrected.

**ISSN: 2580-409X (Print) / 2549-6824 (Online)**
**DOI:** https://doi.org/10.29407/intensif.v7i2.20103

## IV. CONCLUSION

The results of this study indicate that Nazief & Adriani and ECS can be modified for Madurese language stemming by adjusting the Madurese word formation rules. The accuracy obtained for the Nazief & Adriani modification is 88.80%. For Prefix, an accuracy of 93.49% is obtained, Suffix is 83.78%, and for Confix is 88.10%. As for the ECS modification, the prefix accuracy was 82.38%, the suffix accuracy was 59.46% and the confix accuracy was 65.80%.

The results of this study indicate that the method is able to read the prefix and the suffix "N-" which in the process undergoes fusion according to the type of anuswara affix, which indicates that the method modification has been able to resolve the morphophonemic in the basic words. This research has not been able to complete confix and reduplication.. In calculating the success of stemming still use the manual method, so in future research it is necessary to add a method of calculating success, understemming or overstemming to the basic words of stemming results.

## REFERENCES

[1] F. L. Fitri Lintang and F. Ulfatun Najicha, “Nilai-Nilai Sila Persatuan Indonesia Dalam Keberagaman Kebudayaan Indonesia,” *J. Glob. Citiz. J. Ilm. Kaji. Pendidik. Kewarganegaraan*, vol. 11, no. 1, pp. 79–85, 2022, doi: 10.33061/jgz.v11i1.7469.

[2] R. Peter and M. S. Simatupang, “Keberagaman Bahasa Dan Budaya Sebagai Kekayaan Bangsa Indonesia,” *Dialekt. J. Bahasa, Sastra Dan Budaya*, vol. 9, no. 1, pp. 96–105, 2022, doi: 10.33541/dia.v9i1.4028.

[3] A. F. Hidayati, “Afiks Nomina Deverbal dalam Kumpulan Cerpen Bahasa Madura,” *Konf. Linguist. Tah. Atma Jaya 19*, pp. 17–20, 2021.

[4] I. Irwiandi and M. Norman, “Proses Morfologis pada Bahasa Madura: Studi pada Mahasiswa Madura di Universitas Trunojoyo,” *AIJER Algazali Int. J. Educ. Res.*, vol. 5, no. 1, pp. 68–75, 2022.

[5] T. Winarti *et al.*, “Penanganan Kasus Overstemming dan Understemming dengan Modifikasi Algoritma Stemming,” *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 6, no. 1, pp. 199–206, 2020, doi: 10.18517/ijaseit.7.5.1705.

[6] A. Andriani, “Morfofonemik Bahasa Indonesia Pada Masyarakat Tutur Bugis Dialek Sidenreng Rappang,” pp. 1–28, 2021, [Online]. Available: http://eprints.unm.ac.id/20489/%0Ahttp://eprints.unm.ac.id/20489/1/ARTIKEL.pdf

[7] N. W. Wardani and P. G. S. Cipta Nugraha, “STEMMING DOKUMEN TEKS BAHASA BALI DENGAN METODE RULE BASE APPROACH,” *JATISI (Jurnal Tek. Inform. dan Sist. Informasi)*, vol. 7, no. 3, pp. 510–521, 2020, doi: 10.35957/jatisi.v7i3.538.

[8] S. Ibrihich, A. Oussous, O. Ibrihich, and M. Esghir, “A Review on recent research in information retrieval,” *Procedia Comput. Sci.*, vol. 201, no. C, pp. 777–782, 2022, doi: 10.1016/j.procs.2022.03.106.

[9] R. Kurniasari, R. Santoso, and A. Prahutama, “ANALISIS KECENDERUNGAN LAPORAN MASYARAKAT PADA ‘LAPORGUB..!’ PROVINSI JAWA TENGAH MENGGUNAKAN TEXT MINING DENGAN FUZZY C-MEANS CLUSTERING,” *J. Gaussian*, vol. 10, no. 4, pp. 544–553, 2021, doi: 10.14710/j.gauss.v10i4.33101.

[10] Z. Muhamad, “Jurnal Teknik Informatika Atmaluhur,” *J. Tek. Inform. Atmaluhur*, vol. 6, no. 1, p. 40, 2018.

[11] S. B. Rossi Hersianie, "Analisa Modifikasi Algoritma Stemming Untuk Kasus Overstemming," *Teknokom*, vol. 3, no. 2, pp. 23–28, 2020, doi: 10.31943/teknokom.v3i2.51.
[12] A. S. Nurul Justina Mahardianingroem, "Utuk Mengurangi Kesalahan Stemming Bahasa," vol. 10, no. 2, pp. 103–112, 2018.
[13] R. Maulidi, "Modifikasi Metode Enhanced Confix Stripping," *Pros. Semin. Nas. FDI*, no. December, pp. 12–15, 2016.
[14] A. Sholihin, F. Solihin, and F. H. Rachman, "Penerapan Modifikasi Metode Enhanced Confix Stripping Stemmer Pada Teks Berbahasa Madura," *J. Sarj. Tek. Inform.*, vol. 2, no. 1, pp. 305–314, 2013.
[15] A. P. Wibawa, F. A. Dwiyanto, I. A. E. Zaeni, R. K. Nurrohman, and A. Afandi, "Stemming javanese affix words using nazief and adriani modifications," *J. Inform.*, vol. 14, no. 1, p. 36, 2020, doi: 10.26555/jifo.v14i1.a17106.
[16] M. A. Nq, L. P. Manik, and D. Widiyatmoko, "Stemming Javanese: Another Adaptation of the Nazief-Adriani Algorithm," *2020 3rd Int. Semin. Res. Inf. Technol. Intell. Syst. ISRITI 2020*, pp. 627–631, 2020, doi: 10.1109/ISRITI51436.2020.9315420.
[17] N. Hidayatullah, Aji Prasetya Wibawa, and Harits Ar Rosyid, "Penerapan ECS Stemmer untuk Modifikasi Nazief & Adriani Berbahasa Jawa," *J. RESTI (Rekayasa Sist. dan Teknol. Informasi)*, vol. 3, no. 3, pp. 343–348, 2019, doi: 10.29207/resti.v3i3.994.
[18] D. Soyusiawaty, A. H. S. Jones, and N. L. Lestariw, "The Stemming Application on Affixed Javanese Words by using Nazief and Adriani Algorithm," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 771, no. 1, 2020, doi: 10.1088/1757-899X/771/1/012026.
[19] M. A. Muchtar *et al.*, "Separation of Basic Words in Angkola Batak Text Documents using Enhanced Confix Stripping Stemmer Case: Mandailing Ethnic," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 648, no. 1, 2019, doi: 10.1088/1757-899X/648/1/012024.
[20] J. Jumadi, D. S. Maylawati, L. D. Pratiwi, and M. A. Ramdhani, "Comparison of Nazief-Adriani and Paice-Husk algorithm for Indonesian text stemming process," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 1098, no. 3, p. 032044, 2021, doi: 10.1088/1757-899x/1098/3/032044.
[21] I. P. M. Wirayasa, I. M. A. Wirawan, and I. M. A. Pradnyana, "Algoritma Bastal: Adaptasi Algoritma Nazief & Adriani Untuk Stemming Teks Bahasa Bali," *J. Nas. Pendidik. Tek. Inform.*, vol. 8, no. 1, p. 60, 2019, doi: 10.23887/janapati.v8i1.13500.
[22] D. Wahyudi, T. Susyanto, and D. Nugroho, "Implementasi Dan Analisis Algoritma Stemming Nazief & Adriani Dan Porter Pada Dokumen Berbahasa Indonesia," *J. Ilm. SINUS*, vol. 15, no. 2, pp. 49–56, 2017, doi: 10.30646/sinus.v15i2.305.
[23] K. N. Lakonawa, S. A. S. Mola, and A. Fanggidae, "Nazief-Adriani Stemmer Dengan Imbuhan Tak Baku Pada Normalisasi Bahasa Percakapan Di Media Sosial," *J. Komput. dan Inform.*, vol. 9, no. 1, pp. 65–73, 2021, doi: 10.35508/jicon.v9i1.3749.
[24] P. Ruriana, "Hubungan Kekerabatan Bahasa Jawa Dan Madura," *Kandai*, vol. 14, no. 1, p. 15, 2018, doi: 10.26499/jk.v14i1.512.
[25] M. H. Effendy, "Problematika Periodisasi Ejaan Bahasa Madura Dalam Perspektif Praktisi Madura," *Okara*, vol. 2, pp. 24–34, 2013.
[26] N. W. Wardani and P. G. S. C. Nugraha, "Stemming Teks Bahasa Bali dengan Algoritma Enhanced Confix Stripping," *Int. J. Nat. Sci. Eng.*, vol. 4, no. 3, p. 103, 2020, doi: 10.23887/ijnse.v4i3.30309.
[27] H. Mukhtar, J. Al Amien, and M. A. Rucyat, "Filtering Spam Email menggunakan Algoritma Naïve Bayes," *J. CoSciTech (Computer Sci. Inf. Technol.*, vol. 3, no. 1, pp. 9–19, 2022, doi: 10.37859/coscitech.v3i1.3652.
[28] S. Firman, W. Desena, A. Wibowo, M. I. Komputer, and U. B. Luhur, "Penerapan Algoritma Stemming Nazief & Adriani Pada Proses Klasterisasi Berita Berdasarkan Tematik Pada Laman (Web) Direktorat Jenderal HAM Menggunakan Rapidminer," *Syntax J. Inform.*, vol. 11, no. 02, pp. 10–21, 2022.

**ISSN: 2580-409X (Print) / 2549-6824 (Online)**
**DOI:** https://doi.org/10.29407/intensif.v7i2.20103

[29] W. Hidayat, E. Utami, and A. D. Hartanto, "Effect of Stemming Nazief Adriani on the Ratcliff/Obershelp algorithm in identifying level of similarity between slang and formal words," *2020 3rd Int. Conf. Inf. Commun. Technol. ICOIACT 2020*, pp. 22–27, 2020, doi: 10.1109/ICOIACT50329.2020.9331973.

[30] G. Septian, A. Susanto, and G. F. Shidik, "Indonesian news classification based on NaBaNA," *Proc. - 2017 Int. Semin. Appl. Technol. Inf. Commun. Empower. Technol. a Better Hum. Life, iSemantic 2017*, vol. 2018-Janua, pp. 175–180, 2017, doi: 10.1109/ISEMANTIC.2017.8251865.