Accredited Ranking SINTA 2
Decree of the Director General of Higher Education, Research and Technology, No. 158/E/KPT/2021
Validity period from Volume 5 Number 2 of 2021 to Volume 10 Number 1 of 2026

Published online on: **http://jurnal.iaii.or.id**

JURNAL RESTI
(Rekayasa Sistem dan Teknologi Informasi)
Vol. 7 No. 1 (2023) 1341 - 1347 ISSN Media Electronic: 2580-0760

ANoM STEMMER: Nazief & Andriani Modification for Maduresse Stemming

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

Abstract

Madurese is one of the regional languages in Indonesia. This is a cultural property that needs to be preserved. With various uniqueness and word formation rules, the Madurese language can be used in Information Retrieval, namely stemming. The Madurese language has a close relationship with the Javanese language, in several studies the stemming method is often used, such as the modification of the Nazief & Adriani method which has good performance for the Javanese language, but there has never been any research on the Madurese language and it has not been proven successful. Previous studies also have not used morphophonemic rules that influence word formation in Madurese. Therefore this research was developed by modifying Nazief & Adriani's algorithm for Madurese language based on Madurese language morphology by removing affixes, namely ter-ater (prefix), panoteng (suffix), and morphophonemic rules. Corpus uses 1000 words from the Madurese language dictionary which have received affixes. The accuracy of the algorithm is 89% with 890 words that match, the prefix has an accuracy of 93.81%, the suffix has an accuracy of 83.78% and the confix has 80.07%. As for the overall performance, it produces an accuracy of 89.0% with an error rate of 11%. the understemming is found in 104 words and overstemming in 6 words. the time it takes to compile is 31.31 seconds.

Keywords: stemming; morphophonemic; understemming; overstemming; madurese language

1. Introduction

The Madurese language is one of the regional languages in Indonesia. It is used as an everyday language on the island of Madura and other East Java horseshoe areas such as Pasurauan, Probolinggo, Lumajang, Jember, Sitobondo, Bondowoso, and Banyuwangi.

Morphology in the Madurese language has much uniqueness. Morphology is a field of science that studies language units as grammatical units. Meanwhile, the morphological process is the process of word formation originating from a basic form through a process of affixes, repetitions, and composition [1]. There are several kinds of affixes (embuwen) in Madurese, namely prefixes (ter-ater), suffixes (panoteng), infix (sesselan), reduplication (oca' rangkep) [2].

Morphophonemics is a field of science that examines sound changes or phoneme changes caused by morphological processes, both affixation, reduplication, and compositional processes [3].

The rules for forming affixes in the Madurese language can be used to develop Information Retrieval applications. Information Retrieval is a method for presenting, storing, and searching large data sets for information mining purposes and access to find relevant results that meet user needs in response to user requests [4]. Research on natural language processing has become a topic of interest in the fields of artificial intelligence and computational linguistics. An important part of natural language processing is the actual process of identifying and removing morphologically inflected words, commonly called stemming, so that words with the same root can be grouped together.

Stemming is a process of determining basic words by removing the affixes contained in these words[5]. Research on the stemming of the Madurese language is still very small. Stemming algorithm used for Madurese [2], [6]. The Madurese language is related to the Javanese language, namely West Austonesia. Two languages or more that come from the same ancestral language have good correspondence at the phonological

Accepted: 11-05-2023 | Received in revised: 24-11-2023 | Published: 12-12-2023

and lexical level [7]. The Nazief & Adriani algorithm is a basic word search algorithm in Indonesian using Indonesian morphological rules by removing prefixes and word endings according to the Indonesian dictionary [8]. Nazief & Adriani's algorithm is often used for stemming the Java language, as was done in research [9]–[11], while other languages are used in research [12]–[16]. In the stemming process will find understemming, namely the reduction of words that are too few than they should be understemming and overstemming [17], namely the occurrence of fragmentation of words that are more excessive than they should be so that they have different meanings [18].

Therefore, this study uses a modified Nazief & Adriani algorithm for stemming the Madurese language by taking into account morphophonemic rules that have not been used in previous studies. The benefits of this stemming can be used to build search engines, preprocessing Madurese language documents, and other search systems without changing the accents on words so that the pronunciation of words doesn't change, and the meaning of words doesn't change according to the latest Madurese spelling[19].

2. Research Methods

The flow of this research methodology can be seen in Figure 1.

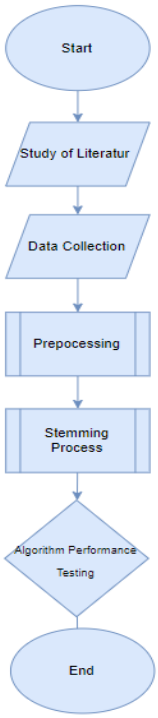

Figure 1. Flowchart of research methodology

2.1 Study of Literature

At this stage, do a literature search and study theories related to the research topic. theses, books, and articles on the Nazief & Adriani algorithm, as well as journals published over the last five years [11].

2.2 Data Collection

This study uses 1000 Madurese words, with 2295 basic words derived from the VI edition of the Contemporary Indonesian Dictionary by Muhri, S.Pd, M.A. These basic words will be input in this research.

2.3 Preprocessing

In this stage, changing the form of unstructured text becomes structured. The steps are: Filtering (It is taking the letters of the alphabet in the Madurese language by changing the accented alphabet to the regular letters of the alphabet); Case Folding (Converts all letters in the word to lowercase. What is processed is the letter "a" to the letter "z.").

2.4 Stemming Process

Three basic principles in stemming Nazief & Adriani[20] :

Eliminate affixes according to the morphological rules of the Indonesian language. There are three kinds of affixes in Indonesian:

Inflectional Endings:
Particle (P) {"-lah " , "-pun", "- tah ", "- kah"}
Pronouns possessive (PP) {“-me”, “-mu”, “-nya”}

Derivation suffixes (DS) {"-i", "- ke ", "-an" }

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

In this study there are two modifications used, the first is to eliminate the particle check stage because in Madura language writing particles are not attached to other words, the second is to remove the check for prohibited affixes.

Modified by Nazief & Adriani for Madurese:

Eliminate affixes according to the morphological rules of the Indonesian language. There are three kinds of affixes in Indonesian:

Inflectional Endings:
Pronouns possessive (PP) {“-na”}
Derivation suffixes (DS) {"-a," "-e," "-na," "-ana," "-an," "-aghi,"}
Derivation prefixes (DP) :
DP morphology: {"nya-," "ma-," "nga-," "-na,"}

Normal DP: {" a -", " ta -", “ma-” “ka-”, “sa-”, “pa-”, “pe-”, “pre-”, “nga-”, “ -e", "-epa", "-eka"}

DOI: https://doi.org/10.29207/resti.v7i6.5086

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

Based on the basic principles of Nazief & Adriani, this research is modified for Madurese. Figure 2 explains the pseudocode on the flowchart

```
1. Algorithm (Stemming to decide the root word from the affixes in Madurese)
2. Declaration :
3. Root word: string
4. Result: string
5. Description :
6. Start
7. Read (affixed Madurese word)
8. Check the root word dictionary
9. Write "kamus kata dasar" ("root word dictionary")
10. If "root word is found" then
11. Print "root word"
12. Else If "root word is not found" then
13. Write "Jalankan Proses Stemming"
("Run the process of Nazief and Adriani algorithm")
14. End.
```

Figure 2. Pseudo Code of System Flowchart

Figure 3 explains the pseudocode Stemming Root By Using Nazief and Adriani

```
1. Algorithm (Stemming root word by using Nazief and Adriani algorithm)
2. Declaration :
3. Root word : string
4. Result : string
5. Description:
6. Start
7. Read (Madurese affixed word)
8. Process of Nazief and Adriani Algorithm
9. Write "Hapus (Delete) Inflectional suffixes"
10. Write "Hapus (Delete) inflectional possessive pronoun suffixes"
11. Write "Hapus (Delete) Derivational Suffix"
12. If "root word is found" then
13. Print "root word"
14. Else If "root word is not found" then
15. Write "Hapus (Delete) Derivational Prefix"
16. If "root word is found" then
17. Print "root word"
18. Else If "root word is not found" then
19. Write "Recording"
20. Print "root word"
21. End
```

Figure 3. Pseudo Code Stemming Root by Using Nazief and Adriani

The rules for word fragments in Madurese language affixes are seen in Table 1.

Table 1. Rule of the Decapitation Affix for the Madurese Language

ID	Word Formats	Word Fragments	Example words
1	eA	eA	eatorè, ecokor, etapok
2	a A	a A	aghâbây, alonca', akemmor
3	taA	ta-A	tabâca, talebat, tapale'
4	maA	ma-A	mataber, maalos, mapotè
5	kaA	ka -A	kaator, katello, kasebhut
6	saA	sa -A	saalam, saarè, sakampong
7	paA	pa-A | pa-nV	paloros, pakonèng, patao
8	komaC	koma-C	komalancang
9	kameC	kame-C	kamèporon
10	kapeC	kape-C	kapèderreng
11	peA	pe-A	pètodhu, pètolong
12	an → nV	n-tV | n-dV	namen, noghel, notop
13	any → ny-nyV	ny-sV | ny-cV | ny-jV	nyate, nyoro, nyotok
14	am → mV	m-pV | m-bV	mancèng, maghâr, maca, mabâ
15	ang → ngV	ng-V | ng-kV|ng-gV	ngoca", ngobâ, ngakan, ngamar
16	panC	pan-C	panjâi"
17	pamC	pam-C	pambâjâr
18	pangC	Pang-C	panglèpor
19	Aè	A-è	toraè, kacaè, pèntaè
20	Vwi	V-wi	toghuwi, tabbhuwi
21	Vne	V-ne	lakonè, nemmonè, astanè
22	Aa	A-a	èntara, ajârâ, toju"â
23	Aan	A-an	pèkkèran, kakanan
24	Vyan	V-yan	belliyân, gengsèyan
25	Vwan	V-wan	kalakowan, robbuwan
26	Aen	A-en	pèttengèn, kacèlleben
27	Vwen	V-wen	kabiruwen,
28	Vna	V-na	robbuna, macana
29	CCa	C-Ca	kaloarra, kasokanna
30	Vana	V-ana	mola"âna, mossa"ana
31	CCana	C-Cana	majârana, nolèsana, malessana
32	Vwana	V-wana	noguwana, ngataowana
33	Vyana	V-yana	mèlèyana
34	Vaghi	V-aghi	sala"aghi, baca"aghi
35	Vwaghi	V-waghi	kokowaghi, nopowaghi
36	Vyaghi	V-yaghi	gâjiyaghi, pèlèyaghi
37	Aepon	A-epon	rabuepon, soroepon, toguepon
38	CCepon	C-Cepon	pèccottepon,

C is a consonant, V is a vowel, A is a vowel or consonant. P is a particle or fragment of a word.

2.5 Algorithm Performance Testing

Evaluation is calculated by dividing the number of correct stemming words by the total number of words multiplied by 100%. Meanwhile, the error rate is calculated by subtracting 100% from the results of accuracy .

3. Results and Discussions

This study uses python in the development of the model. The input of this model is an affixed Madurese word. There are 1000 words with words containing 631 prefix words, 74 suffix words, and 293 confix words as seen in Figure 4.

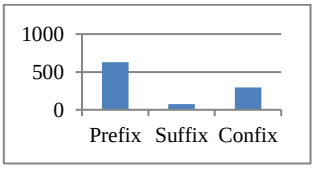

Figure 4. Graph of Total Word

The data and their meaning are explained in Table 2.

DOI: https://doi.org/10.29207/resti.v7i6.5086

Table 2. Raw Data

ID	word	Meaning	
		Indonesian	English
1	aata’	Beratap	roofed
2	abâddhâ	Berwadah	have a container
3	abâgi	Berbagi	share
4	abâi’	Berbiji	seed
5	abâjâng	Melaksanakan Sholat	do Prayer
6	abâkto	mempunyai Waktu	have a time
7	abâlâ	Memberitahukan	tell
8	abâlânjhâ	Berbelanja	shop
9	abâles	membalas	replay
10	abâli	kembali	return
11	abâli’	Berbalik	turn around
12	ngoto’è	Membisiki	whispered
...	...	...	
1000	Tabhâligghâ	kebalikan	reverse

3.1 Preprocessing

The results of this stage can be seen in Table 3

Table 3. Result of Prepocessing

ID	word	Results
1	aata’	aata'
2	abâddhâ	abaddha
3	abâgi	abagi
4	abâi’	abai’
5	abâjâng	abajang
6	abâkto	abakto
7	abâlâ	abala
8	abâlânjhâ	abalanjha
9	abâles	abales
10	abâli	abali
11	abâli’	abali’
12	abângbâng	abangbang
...	...	...
1000	Tabhâligghâ	tabhaliggha

3.2 Stemming Result

Words that have been preprocessed are then stemmed, the result is 3 statuses, namely appropriate, understemming and overstemming. The results of the stemming process can be seen in Table 4.

Table 4. Result of Stemming

ID	word	Stemming	Status
1	abukka’	bukka’	correct
2	abâtek	bâtek	correct
3	abato	bâto	correct
4	abâto’	bâto’	correct
5	abâu	bâu	correct
6	abeddhâ’	bedḍhâ’	correct
7	abegghâ	abek	Overstemming
8	abelli	belli	correct
9	aberri’	berri’	correct
10	abhâjâng	bhâjâng	correct
11	abhâlâ	bhâlâ	correct
12	abhâlik	abhâlik	Understemming
...	...	...	...
1000	Tabhalighâ	bhalik	correct

3.3 Algorithm Performance Testing

Algorithm performance testing is done by calculating accuracy using Formulas 1 and 2[21].

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

$$Error\ rate = 100 - Accuracy \quad (2)$$

From testing using Madurese words, the developed algorithm can find basic words that have prefixes, suffixes, and confixes . However, there are some words that experience understemming or overstemming.

3.4 Discussions

From testing using Madurese words, the developed algorithm can find basic words that have prefixes, suffixes, and confixes . However, there are some words that experience understemming or overstemming. Table 5 is tabel of words that experience overstemming.

Table 5. Overstemming Word

ID	Word	Stemming
1	abârna	abâr
2	abegghâ	abek
3	apèsa	apes
4	ngallè	kallè
5	masana	mas
6	ngennengngè	enneng

The error is like the word "abeggha" the stemming result is "abek". The proper division is “a-bhegghâ”. This happens because the last letter "â" at the end of a word is read as a final affix so that the letter "ggh" which is in front of the letter "a" is replaced with the letter "k" according to the rules so that overstemming occurs.

Errors also occur in the word "abârna" the stemming result is "abâr". The proper sentence fragment is “a-bârna”. This happens because the last letter "nâ" is read as a suffix so that overstemming occurs.

The same error occurs with the word "ngallè" the stemming result is "kalle". The correct word separation is “nga-alle”. This happens because the last letter "ngâ" is immediately changed to the letter "k".

Overstemming also on the word "apèsa". The suffix should be "a-pèsa". In this process, the letter "a" at the end of a word is read as a suffix so that the letter "a" at the end of the word is removed to produce the word "apès".

The next mistake occurred in the word "masana" which resulted in the word "mas", the word that should have been broken was "masa-na". The letter "ana" at the end of a word is read as the suffix "-ana" so that overstemming occurs.

DOI: https://doi.org/10.29207/resti.v7i6.5086

The word "ngennengngè" also has the wrong spelling, it should be "kenneng", but it is changed to "enneng". This happens because the "nge" in front is considered a suffix "ng" so removing it should be changed to "k". Table 6 is tabel of words that experience understemming.

Table 6 Understemming Word

ID	Word	Stemming
1	abhâlik	abhâlik
2	abiluk	abiluk
3	adhârât	adhârât
4	adhungngèng	adhungngèng
5	adhunnynya	adhunnynya
6	aegghât	aegghât
7	aghellâng	aghellâng
8	ajelling	ajelling
9	ajhudu	ajhudu
.....	akapo’	akapo’
104	akèka	akèka

Understemming occurs in the word "abhâlik" because there is no stemming . The resulting word should be the word " bhalik". Also the word "adhârât" should produce "dhârât". Then the word "aghellâng", should form the word "ghellâng". Of all the examples of these words, stemming does not occur, so the resulting word is the initial word. The table 7 of stemming trial results.

Table 7. Result of Trial

ID	Affix	Correct	Over stemming	Understemming	accuracy
1	Prefix	591	3	36	93,81%
2	Suffix	62	1	11	83,78%
3	Confix	237	2	57	80,07%
4	All (Prefix, Suffix,Confix)	890	6	104	89,00%

Table 8 is the result of calculating the error rate from the experiment

Table 8. Result of Error Rate

ID	Affix	accuracy	Error Rate
1	Prefix	93,81%	6,35%
2	Suffix	83,78%	16,22%
3	Confix	80,07%	19,59%
4	All (Prefix, Suffix,Confix)	89,00%	11,00%

This accuracy value indicates that Nazief & Adriani can be used for stemming the Madurese language by making several rule modifications according to the morphology of the Madurese language. The highest success rate is found in the prefix, the lowest is in confix. This is caused by the lack of rules for hyphenation in words that contain Confix.

The use of morphophonemic in this study can produce basic words that have changed form after getting affixes, such as the smelting of consonants from "ng" to "k". This has not been done in [10]so that the word "Nithik" fails to become "thihik"

In contrast to [22] using 219 Javanese words, the accuracy is also greater, namely 93.6%. there are more data is used than that study, namely, previously 400 words were also the same as a research [7] while for this study 1000 words.

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.

4. Conclusion

The results of this study indicate that Nazief & Adriani can be modified for Madurese language stemming by adjusting the word formation rules in Madurese. The accuracy obtained is 89%. For Prefix, an accuracy of 93% is obtained, Suffix is 83.78% and for Confix is 80.07% and error rate is 11%. Compiling this method also takes a fairly fast time, namely 31.31 seconds for 1000 words,.

This study has used morphophonemic rules such as the word "ngotore" becomes "kotor" and the word "nolèsaghi" becomes "tolès" and the word "nyâjhânè" becomes "jajhân". This study also maintains the accent on the alphabet so as not to change the pronunciation of the word so as not to change the meaning of the word.

Because this research has not been able to complete the infix and repetition of Madurese words, in the future you can continue this research by adding Madurese language rules for infix and word repetition in Madurese, as well as adding rules that can reduce overstemming and understemming in decapitation. the word prefix, suffix and confix. It is also necessary to add an algorithm to calculate accuracy so that it does not manually determine the stemming results obtained.

References

[1] I. Irwiadi and M. Norman Antono, “Proses Morfologis pada Bahasa Madura: Studi pada Mahasiswa Madura di Universitas Trunojoyo,” 2019.

[2] F. H. Rachman, N. Ifada, S. Wahyuni, G. D. Ramadani, and A. Pawitra, “ModifiedECS (mECS) Algorithm for Madurese-Indonesian Rule-Based Machine Translation,” in *2022 International Conference of Science and Information Technology in Smart Administration (ICSINTESA)*, IEEE, Nov. 2022, pp. 51–56. doi: 10.1109/ICSINTESA56431.2022.10041470.

[3] A. Andriani, “MORFOFONEMIK BAHASA INDONESIA PADA MASYARAKAT TUTUR BUGIS DIALEK SIDENRENG RAPPANG,” 2021. Accessed: Nov. 12, 2023. [Online]. Available: http://eprints.unm.ac.id/20489/1/ARTIKEL.pdf

[4] S. Ibrihich, A. Oussous, O. Ibrihich, and M. Esghir, “A Review on recent research in information retrieval,” in *Procedia Computer Science*, Elsevier B.V., 2022, pp. 777–782. doi: 10.1016/j.procs.2022.03.106.

[5] Y. K. Paskahningrum, E. Utami, and A. Yaqin, “A Systematic Literature Review of Stemming in Non-Formal Indonesian

DOI: https://doi.org/10.29207/resti.v7i6.5086

Language," *Int J Innov Sci Res Technol*, vol. 8, no. 1, 2023, doi: 10.5281/zenodo.7547482.

[6] M. A. Nq, L. P. Manik, and D. Widiyatmoko, "Stemming Javanese: Another Adaptation of the Nazief-Adriani Algorithm," in *2020 3rd International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)*, IEEE, Dec. 2020, pp. 627–631. doi: 10.1109/ISRITI51436.2020.9315420.

[7] P. Ruriana, "Hubungan Kekerabatan Bahasa Jawa Dan Madura," *Kandai*, vol. 14, no. 1, p. 15, Jul. 2018, doi: 10.26499/jk.v14i1.512.

[8] 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," in *2020 3rd International Conference on Information and Communications Technology, ICOIACT 2020*, Institute of Electrical and Electronics Engineers Inc., Nov. 2020, pp. 22–27. doi: 10.1109/ICOIACT50329.2020.9331973.

[9] 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," *Jurnal Informatika*, vol. 14, no. 1, p. 36, Jan. 2020, doi: 10.26555/jifo.v14i1.a17106.

[10] N. Hidayatullah, A. Prasetya Wibawa, and H. A. Rosyid, "Terakreditasi SINTA Peringkat 2 Penerapan ECS Stemmer untuk Modifikasi Nazief & Adriani Berbahasa Jawa," *masa berlaku mulai*, vol. 1, no. 3, pp. 343–348, 2019.

[11] N. W. Wardani and P. G. S. C. Nugraha, "Stemming Teks Bahasa Bali dengan Algoritma Enhanced Confix Stripping," *International Journal of Natural Science and Engineering*, vol. 4, no. 3, pp. 103–113, Dec. 2020, doi: 10.23887/ijnse.v4i3.30309.

[12] 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, Mar. 2021, doi: 10.1088/1757-899x/1098/3/032044.

[13] I. Putu *et al.*, "Algoritma Bastal: Adaptasi Algoritma Nazief & Adriani Untuk Stemming Teks Bahasa Bali," 2019.

[14] D. Wahyudi, T. Susyanto, D. Nugroho, P. Studi Teknik Informatika, S. Sinar Nusantara Surakarta, and P. Studi Sistem Informasi, "Implementasi Dan Analisis Algoritma Stemming Nazief & Adriani Dan Porter Pada Dokumen Berbahasa Indonesia," *jurnal ilmiah SINUS*, 2018.

[15] K. N. Lakonawa, S. A. S. Mola, and A. Fanggidae, "Nazief-Adriani Stemmer Dengan Imbuhan Tak Baku Pada Normalisasi Bahasa Percakapan Di Media Sosial," *Jurnal Komputer dan Informatika*, vol. 9, no. 1, pp. 65–73, Mar. 2021, doi: 10.35508/jicon.v9i1.3749.

[16] 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," 2022.

[17] N. Justina, M. Verdaningroem, and A. Saifudin, "Penerapan Kamus Dasar Pada Algoritma Porter Untuk Mengurangi Kesalahan Stemming Bahasa Indonesia," 2018, doi: 10.24853/jurtek.10.2.103-112.

[18] S. Betha and R. Hersianie, "Analisa Modifikasi Algoritma Stemming Untuk Kasus Overstemming," vol. 3, no. 2, 2020.

[19] M. Hafid, E. Dosen, J. Tarbiyah, and S. Pamekasan, "Problematika Periodisasi Ejaan Bahasa Madura Dalam Perspektif Praktisi Madura."

[20] M. A. Nq, L. P. Manik, and D. Widiyatmoko, "Stemming Javanese: Another Adaptation of the Nazief-Adriani Algorithm," in *2020 3rd International Seminar on Research of Information Technology and Intelligent Systems, ISRITI 2020*, Institute of Electrical and Electronics Engineers Inc., Dec. 2020, pp. 627–631. doi: 10.1109/ISRITI51436.2020.9315420.

[21] A. Yaqin, M. Rahardi, and F. F. Abdulloh, "Accuracy Enhancement of Prediction Method using SMOTE for Early Prediction Student's Graduation in XYZ University," *International Journal of Advanced Computer Science and Applications*, vol. 13, no. 6, pp. 418–424, 2022, doi: 10.14569/IJACSA.2022.0130652.

[22] D. Soyusiawaty, A. H. S. Jones, and N. L. Lestariw, "The Stemming Application on Affixed Javanese Words by using Nazief and Adriani Algorithm," in *IOP Conference Series: Materials Science and Engineering*, Institute of Physics Publishing, Mar. 2020. doi: 10.1088/1757-899X/771/1/012026.

DOI: https://doi.org/10.29207/resti.v7i6.5086