Comparison of Effectiveness of Stemming Algorithms in Indonesian Documents

by Dyah Mustikasari, Ida Widaningrum, Rizal Arifin, Wahyu Henggal Eka Putri

Comparison of Effectiveness of Stemming Algorithms in Indonesian Documents

Dyah Mustikasari[1] Ida Widaningrum[2,*] Rizal Arifin[3] Wahyu Henggal Eka Putri[4]

[1,2,3,4] *Department of Informatics, Universitas Muhammadiyah Ponorogo, 63471, Indonesia*
[*] *Corresponding author. Email: iwidaningrum@umpo.ac.id*

ABSTRACT

Stemming is a process to determine basic word with some rules. In Bahasa Indonesia, the way is to eliminate prefixes, infixes, suffixes, or combination of prefixes and suffixes in derivative words. Several stemming algorithms for Bahasa Indonesia have been developed. But their effectiveness has not been studied. In this study, these three stemming algorithms will be compared. We used 900 affixes to conduct the comparison. Each word is searched for their basic words using the three algorithms. The basic word resulted then referred to KBBI or Indonesian dictionary to see whether they are right. Comparison process of stemming show that Sastrawi's could do the best stemming that 95,2% of the affix words tested could be root words. The Nazief & Adriani Algorithm resulted 92,4%, while Arifin Setiono's finished at 89%. It could state that Arifin Setiono's needs a lot of improvement because many affixed words could not return to the root word.

Keywords: *Effectiveness, Stemming, Indonesian, Document.*

1. INTRODUCTION

Stemming is a process to return affix word to its basic or root word using prescribed rules. The way is by removing the prefix, infix (insertion), suffix, and confix (combination of prefix and suffix) from affix words. Stemming is important part in Information Retrieval for web search, document clustering in term of decreasing the number of different indexes of a document, and translation [1]. A good stemmer or algorithm will return affix word to the root word correctly. In Bahasa, the prefix are *pe-, me-, ber-, di-, ke-, ter-,* and the confix are *ke-an, ber-an, pe-an, se–nya*. For example, word *baca* that means 'reading', could be given prefix *me-* that change to *membaca,* not *mebaca*. This causes there are rules for changing phonemes in the formation of affixed words in Bahasa [2]. Another example, the word *pukul* will be *memukul* if it added prefix *me-*, not *mempukul*. The phoneme /p/ in the word *pukul* is disappeared. However, when the word *proses* is added prefix *me-*, it become *memproses*, not *memroses*. In this case, the phoneme /p/ in the word *proses* is maintained and not disappeared.

Some stemming rules have been developed for Bahasa Indonesia, such as the Nazief & Adriani Algorithm (1996) which was later accomplished by Jelita Asian (2005) [1] , Vega Bressan [1], Arifin Setiono's Algorithm (2002) [3], and the latest is Sastrawi's Stemmer [4].

The Nazief & Adriani algorithm was proposed by Bobby Nazief and Mirna Adriani [5]. This algorithm uses morphological rules in Bahasa. They are collected together and encapsulated in the allowed affixes and the disallowed affixes [1]. Arifin Setiono's algorithm has a similar process to Nazief & Adriani's algorithm but it is assumed that a word has two prefixes and three suffixes [3]. While, Sastrawi Algorithm applies an algorithm based on Nazief-Adriani, then improved by the CS (Confix Stripping) Algorithm, and enhanced by the ECS (Enhanced Confix Stripping) algorithm, and improved again by Modified ECS.

With these various stemming methods, it has not been known which algorithm is best for returning affixed words to the root words in Bahasa, yet. Some comparative studies have been done before. One of the comparison that has been conducted is [6] that compared stemming Algorithms among Nazief-Adriani, Arifin-Setiono, Tala, and Vega. Another comparison of stemming algorithm conducted by [3] that compare between Porter and Arifin-Setiono [7]. Comparison of Nazief-Adriani and Idris also conducted by [8]. A comparative study conducted by [5] tested among some algorithm, namely porter confix striping, Nazief, Arifin, Fadillah, Asian, Enhanced confix stripping, dan Arifiyanti [9]. Sastrawi Algorithms has not been compared in the previous studies. So, this research will conduct comparison among, Nazief-Adriani Algorithms, Arifin-Setiono Algorithms and Sastrawi.

2. METHOD

2.1. Nazief-Adriani

Nazief & Adriani Algorithms uses root word dictionary created by Bobby Nazief and Mirna Adriani. The algorithms have the following steps:

1. Checking if the word is in the root word dictionary. If it is found, it is considered the root word, and then the process is stopped.
2. Removing inflection suffixes ("-lah", "-kah", "-nya", "-mu", or "- ku"). If it is done, then the suffixes is a particle of “-kah” and “-lah”, the step is repeated for removing the suffixes of possessive pronoun (“-ku”, “-mu”, “-nya”)
3. Deleting suffix ("-i" or "-an"). If the root word is found, the process continues to 4. If it did not, then the step continues to 3a.
   a. If the last letter is "-k", the letter "-k" is deleted and do the step 4. But, if the root word is still not found, then continue to step 3b.
   b. The deleted suffix ("-i", "- an" and also "-kan") are restored, then it continue to step 4
4. Removing prefixes ("be -", "di-", "me -", "ke -", "pe -", "te-" and "se-"). If the word matches the root word dictionary, the process can be stopped. But, if the root word has not been found, the word is recoded. This process can be stopped if:
   a. Incorrect combination of prefix and suffix.
   b. The prefix detected with the previously omitted prefix is the same.
   c. Removing three prefixes.
5. If all of the steps has been done but the root word is not resulted, then the algorithm will return the word to the way it was before stemming.

The Nazief & Adriani was then accomplished by Asian [1]. The improvements of Jelita Asian were:

1. completing the dictionary,
2. adding rules for plurals (such as *buku-buku*, *berbalasan-balasan*, and so on),
3. adding prefix and suffix such as *-pun*, modifying condition for prefix *ter-, pe-, mem-, meng-*
4. changing the stemming sequence of the affix words with prefix *ber-* and suffix *-lah*, prefix *ber-* and suffix *-an*, prefix *me-* and suffix *-i,* prefix *di-* and suffix *-i*, prefix *pe-* and suffix *-i,* and prefix *ter-* and suffix *-i,* to delete the prefix first and then the suffix.

2.2. Arifin-Setiono Algorithm

In 2002, Agus Zainal Arifin and Ari Novan Setiono proposed some rules for returning affix word to the root word, that later known as Arifin-Setiono Algorithm [3]. This algorithm assumed that every word has two prefixes and three suffixes, then conforms the following pattern:

AW 1 + AW 2 + KD + AK 3 + AK 2 + AK 1

which are:

AW 1	= Prefix 1
AW 2	= Prefix 2
KD	= root words
AK 1	= Suffix 3
AK 2	= Suffix 2
AK 3	= Suffix 1

If a word has prefix or suffix less than that, then for empty prefix it is marked with x and xx for empty suffix. Stemming are conducted in the following sequence:

a. first, removing AW 1, then the results are stored at p1
b. then removing AW 2, so the results are stored at p2
c. then removing AK 3, so the results are stored at s1
d. then removing AK 2, and the results are stored at s2
e. finally, removing AK 1, then the results are stored at s3

The cutting results of each sequence are matched with the dictionary whether it has returned to the root word. If it has returned to the root word, then stemming is stopped, otherwise the process is continued. If the root word has not been found in the dictionary until the end of stemming sequence, then the result is combined with affix using the following 12 configurations:

a. KD
b. KD + AK 3
c. KD + AK 3 + AK 2
d. KD + AK 3 + AK 2 + AK 1

e. AW 1 + AW 2 + KD
f. AW 1 + AW 2 + KD + AK 3
g. AW 1+ AW 2 + KD + AK 3 + AK 2
h. AW 1+ AW 2 + KD + AK 3 + AK 2 +AK1
i. AW 2 + KD
j. AW 2 + KD + AK 3
k. AW 2 + KD + AK 3 + AK 2
l. AW 2 + KD + AK 3 + AK 2 + AK 1

These rules are written in python that could be found in the following link: http://tiny.cc/stemmingarifinsetiono

2.3. *Sastrawi*

Sastrawi is actually a stemmer library. This library is available on a source code provider site and can be accessed at the link https://github.com/sastrawi/sastrawi. [4] reviewed that this library is based on research from [1][5][10]. It wrote at the site that the stemming process using this stemmer relies heavily on the root word dictionary. It uses basic word dictionary from kateglo.com with minor changes. The rules of Sastrawi stemmer are as follows :

a. First is checking whether the word will be stemmed is on the dictionary of root words or not. If it exists, so the process will stop at this step.

b. If the word is not in the dictionary, that means it is an affix word, then it is eliminated its suffix *-lah, -kah, -ku, -mu, -nya, -lah, -kah, -tah* or *-pun*.

c. Removing derivative affixes *-i, -kan, -an,* then deleting *be-, di-, ke-, me, pe-, se-* and *te-*.

d. If the root word resulted from the steps before is not found in the dictionary, then it is checked whether the word is included in the ambiguous table in the last column or not.

e. At last, when all the above steps fail, the algorithm returns the word to its original word.

All above algorithms use root word dictionary that could be accessed at http://tiny.cc/rootwords.

2.4. *The Comparison*

We started by collecting data tested. There are 900 affixes in Bahasa to test that could be accessed at link http://tiny.cc/dataset_stemming. Tabel 1 illustrated 50 affixes used for stemming by Sastrawi , Nazief-Adriani algorithm, and Arifin Setiono's algorithm.

Table 1. Example of Affixed Words

Affixes				
berantai	memakai	ajaran	kebijakan	berkenalan
bajakan	mengunjungi	dinyatakan	siapkah	perampasan
berkenalan	pegangan	dirampas	terasing	perenang
berurutan	mengenakan	ditunjukkan	teratur	terimalah
beterbangan	serapan	pemancar	keberagaman	ukuran
bukankah	terimalah	pembentukan	walaupun	ulangan
memangkas	teringat	pemberian	pernikahan	pembangkit
pemadam	terinjak	penggelapan	diperhatikan	kecepatam
mengepalai	teriris	pengikut	perencana	diselami
menikah	terbitan	penjagaan	mengarang	menipis

The words were collected randomly from Indonesian Dictionary. Each word is searched for its root words using these three algorithms. The result from the stemming process were referred to Indonesian Dictionary or Kamus Besar Bahasa Indonesia to check whether it is correct or not. They were also checked manually to make sure it returns to the root word according to the context of the word. The method briefly depicted in Figure 1.

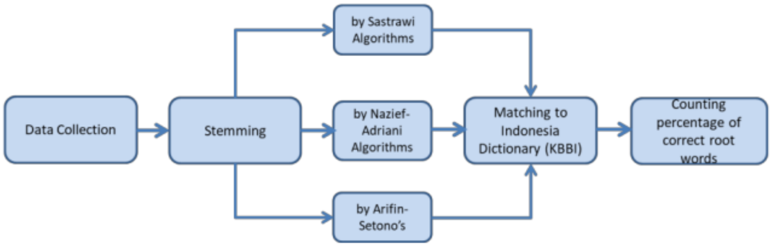
Data Collection
Stemming
by Sastrawi Algorithms
by Nazief-Adriani Algorithms
by Arifin-Setono's
Matching to Indonesia Dictionary (KBBI)
Counting percentage of correct root words

Figure 1 The sequence of research.

3. RESULT AND DISCUSSION

We use 900 affixes to proceed in stemming. Each algorithm has returned the root word with some fault. Sastrawi did 43 mistakes in stemming process or could return 857 of correct root words. Table 2 shows ten of the mistakes.

Table 2. Example of Incorrect Root Words Obtained by Sastrawi

No.	Affixed words	KBBI	Sastrawi
1	berantai	rantai	beranta
2	bajakan	bajakan	baja
3	berkenalan	kenal	nal
4	berurutan	urut	rurut
5	beterbangan	terbang	bangan
6	bukankah	bukan	bukankah
7	memangkas	pangkas	mangkas
8	pemadam	padam	madam
9	mengepalai	kepala	palai
10	menikah	nikah	meni

Nazief-Adriani algorithm is able to return 832 correct affixes according to the KBBI. That means there are 68 mistakes did by Nazief-Adriani in stemming process. Table 3 illustrates some of the mistakes.

Table 3. Example of Incorrect Root Words Obtained by Nazief-Adriani

No.	Affixed words	KBBI	Nazief-Adriani
1	berantai	rantai	anta
2	bajakan	bajakan	baja
3	kebijakan	bijak	bija
4	memakai	pakai	maka
5	mengunjungi	kunjung	unjung
6	bukankah	bukan	bu
7	memangkas	pangkas	mangkas
8	pemadam	padam	madam
9	pegangan	pegang	gang
10	menikah	nikah	meni

There are several words that have identical affixes but come from different root words, for example the word "mengepak" this word can be stemmed to root word “epak” (take the right to something that produces results by paying rent or taxes) and “kepak” (flapping). According to KBBI, if both word are added prefix “me-“, they turn to word “mengepak”. Another example, the words “acau” (talk in one's sleep) and “kacau” (messy), when given prefix “me- “, become word "mengacau". This indicates that stemming process based on checking the root dictionary, returns the affixes to the root word in the first order. So for affix “mengacau” will always return to root word of “acau”. The whole result of stemming proceed by the three algorithms could be found at link http://tiny.cc/result_stemming. From the result, we calculate percentage of correct root words by:

$$\frac{\text{Correct root words}}{\text{data set}} \text{ x } 100\%$$

The correct root words are the result of stemming that are matched with the word in Indonesia Dictionary or KBBI. Table 5 shows the percentage of the three algorithms.

While the affixes that can be returned to the correct root word by Arifin-Setiono’s algorithm are as many as 801. There are 99 root words did not match KBBI. Ten of mistake of Arifin-Setiono’s Algorithm is showed by Table 4.

Table 4. Example of Incorrect Root Words Obtained by Arifin-Setiono

No.	Affixed words	KBBI	Arifin-Setiono
1	berantai	rantai	berantai
2	bajakan	bajakan	baja
3	kebijakan	bijak	bija
4	memakai	pakai	maka
5	berurutan	urut	berurutan
6	bukankah	bukan	bukankah
7	mengenakan	kena	enak
8	pemadam	padam	madam
9	pegangan	pegang	gang
10	serapan	serap	rap

Table 5. Percentage of correct root word of stemming procees

Algorithm	Correct Root Word	Incorrect Root Word	Percentage of correct root word
Sastrawi	857	43	95,2%
Nazief-Adriani	832	68	92,4%
Arifin-Setiono	801	99	89%

4. CONCLUSION

These results indicate that the highest yield of the three stemming algorithms is Sastrawi stemmer, followed by Nazief-Adriani. It also could state that Arifin Setiono Algorithm needs a lot of improvement because many affixed words could not return to the root word.

ACKNOWLEDGMENT

We would like to thank Direktorat Riset dan Pengabdian Masyarakat, Direktorat Jenderal Penguatan Riset dan Pengembangan Kementerian Riset, Teknologi, dan Pendidikan Tinggi Republik Indonesia, Ministry of Research and Technology.

REFERENCES

[1] J. Asian, H. E. Williams, and S. M. M. Tahaghoghi, “Stemming Indonesian,” in *28th Australasian Computer Science Con- ference (ACSC2005)*, 2005, vol. 38.

[2] R. Setiawan, A. Kurniawan, W. Budiharto, and A. Prefix, “Flexible Affix Classification for Stemming Indonesian Language,” in *13th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, 2016.

[3] A. Zainal and A. Novan, “Klasifikasi Dokumen Berita Kejadian Berbahasa Indonesia dengan Algoritma Single Pass Clustering,” in *Proceedings of the Seminar on Intelligent Technology and its Applications (SITIA)*, 2002.

[4] U. Hasanah, T. Astuti, R. Wahyudi, Z. Rifai, and R. A. Pambudi, “An Experimental Study of Text Preprocessing Techniques for Automatic Short Answer Grading in Indonesian,” in *3rd International Conference on Information Technology, Information System and Electrical Engineering (ICITISEE)*, 2018, pp. 230–234.

[5] B. Nazief and M. Adriani, “Confix stripping: Approach to Stemming Algorithm for Bahasa Indonesia,” Jakarta, 1996.

[6] M. S. H. Simarangkir, “Studi Perbandingan Algoritma-Algoritma Stemming untuk Dokumen Teks Bahasa Indonesia,” *J. Infokar*, vol. 1, no. 1, pp. 40–46.

[7] D. Novitasari, “Perbandingan Algoritma Stemming Porter dengan Arifin Setiono untuk Menentukan Tingkat Ketepatan kata Dasar,” vol. 1, no. 2, pp. 120–129, 2016.

[8] A. Prasidhata and K. M. Suryaningrum, “Perbandingan Algoritma Nazief & Adriani Dengan Algoritma Idris Untuk Pencarian Kata Dasar,” *J. Teknol. dan Manaj. Inform.*, vol. 4, no. 1, pp. 1–4, 2018.

[9] A. S. Rizki, “Perbandingan Stemmer Bahasa Indonesia dan Dampaknya pada Penggalian Teks Bahasa Indonesia, Studi Kasus Pengelompokan Keluhan Pelanggan PLN,” Institut Teknologi Sepuluh Nopember, 2017.

[10] A. Z. Arifin and H. T. Mahendra, I Putu Adhi Kerta Ciptaningtyas, “Enhanced Confix Stripping Stemmer and Ants Algorithm for Classifying News Documents in Indonesian Language,” in *The 5th International Conference on Information & Communication Technology and Systems*, 2007, pp. 149–158.

Comparison of Effectiveness of Stemming Algorithms in Indonesian Documents

ORIGINALITY REPORT

8%	5%	8%	4%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	sinbad2.ujaen.es Internet Source	1%
2	"Recommender System using Content Based Filtering for News Portal in Indonesia", International Journal of Recent Technology and Engineering, 2020 Publication	1%
3	I G M Darmawiguna, G A Pradnyana, G S Santyadiputra. "The Development of Integrated Bali Tourism Information Portal using Web Scrapping and Clustering Methods", Journal of Physics: Conference Series, 2019 Publication	1%
4	Sherly Christina, Enny Dwi Oktaviyani, Buyung Famungkas. "Mendeteksi Plagiarism Pada Dokumen Proposal Skripsi Menggunakan Algoritma Jaro Winkler Distance", Jurnal SAINTEKOM, 2018 Publication	1%

5	Handayani Nurina Putri, Herwany Aldrin. "Linking job expectation, career perception, intention to stay: Evidence from generation Y", HOLISTICA – Journal of Business and Public Administration, 2019 Publication	1 %
6	Amalia Amalia, Maya Sylvi Lidya, Andrian Andrian, Elviawaty Muisa Zamzami, Sri Melvani Hardi. "OLCBot: Dissemination Of Interactive Information Related To Indonesia's Omnibus Law With The Implementation of Fuzzy String Matching Algorithm and Sastrawi Stemmer", 2022 6th International Conference on Electrical, Telecommunication and Computer Engineering (ELTICOM), 2022 Publication	1 %
7	eprints.binadarma.ac.id Internet Source	1 %
8	Lois Tweneboa Kodua, Yuchun Xiao, Nana Osae Adjei, Dennis Asante, Bright Okyere Ofosu, David Amankona. "Barriers to green human resources management (GHRM) implementation in developing countries. Evidence from Ghana", Journal of Cleaner Production, 2022 Publication	1 %
9	Anton Yudhana, Abdul Fadlil, Muhamad Rosidin. "Indonesian Words Error Detection	1 %

System using Nazief Adriani Stemmer Algorithm", International Journal of Advanced Computer Science and Applications, 2019

Publication

Exclude quotes	On	Exclude matches	< 1%
Exclude bibliography	On		