PAPER • **OPEN ACCESS**

The Stemming Application on Affixed Javanese Words by using Nazief and Adriani Algorithm

To cite this article: Dewi Soyusiawaty *et al* 2020 *IOP Conf. Ser.: Mater. Sci. Eng.* **771** 012026

IOP Conf. Series: Materials Science and Engineering **771** (2020) 012026 doi:10.1088/1757-899X/771/1/012026

The Stemming Application on Affixed Javanese Words by using Nazief and Adriani Algorithm

Dewi Soyusiawaty[1,*], Anna Hendri Soleliza Jones[1], Nora Novita Lestari[1]

Informatics Engineering, Ahmad Dahlan University, Yogyakarta, Indonesia.

[*]dewi.soyusiawaty@tif.uad.ac.id, annahendri@tif.uad.ac.id

Abstract. The Javanese language has various forms of affixed words. It makes people from other regions find it difficult to understand the Javanese language. One of the ways to understand the Javanese language is by studying and understanding Javanese vocabulary in the Javanese language dictionary or learning directly from ones who master the Javanese language. Both offline and online dictionaries have not been able to translate affixed words. This research used Nazief and Adriani Algorithm to find out the stems/root words in the Javanese language. The objective of this research is to apply the Nazief and Adriani Algorithm in stemming affixed words in the Javanese language. The subject discussed in this research is the application of stemming affixed Javanese words by using Nazief and Adriani algorithm. The stages started with identifying the affixes in the Javanese language. Nazief and Adriani algorithm has five steps those are detecting words that have affixes, cutting words that contain inflection suffixes, derivation suffixes and derivation prefixes, and last recording. The result of this research is a stemming application for affixed Javanese words that applies Nazief and Adriani algorithm method. Based on the accuracy test, it appears that 93.6% of words are accurate from the total of 219 words tested.

1. Introduction

Language is used as a primary means of daily communication, so understanding language is essential. One of the languages that is familiar to Indonesian society is the Javanese language used by societies living in Central Java, East Java, and Special Region of Yogyakarta. The Javanese language has several varieties which are Ngoko Javanese (low), Madyo Javanese (middle), and Kromo (high refined). Quoted from the book "Kawuruh Basa Jawa Pepak", it explains that affixed words in the Javanese language are divided into three those are *tembung lingga oleh ater-ater* (prefixed words), *tembung lingga oleh seselan* (infixed words), and *tembung lingga oleh panambang* (suffixed words) [1]. The way to find the stems or root words in the Javanese language is by using the process of stemming. In Natural Language Processing (NLP), there are several algorithms that can be used in the stage of stemming such as Porter Algorithm, Arifin Setiono Algorithm, Vega Algorithm, Tala Algorithm and Nazief and Adriani Algorithm [2]. This research used Nazief and Adriani algorithm because it uses the best algorithm in the process of stemming [3][4]. Nazief and Adriani algorithm has additional rules for reduplications, prefixes, and suffixes to improve the accuracy of each word. This research is to find out the application of Nazief and Adriani algorithm and its accuracy level in finding Javanese root words.

One of the techniques used in *Natural Language Processing* to restore the form of a word to be its root (root word) is the stemming technique. Stemming is removing prefixes and suffixes from the data of affixed words. *Stemming* is a process or technique used to find out the stem or root word of an affixed word [5]. According to Tala, stemming is used to change the form of a word to be its root

based on the correct morphological structure of the language. There are three methods of stemming in Figure 1, which are [6][7]:

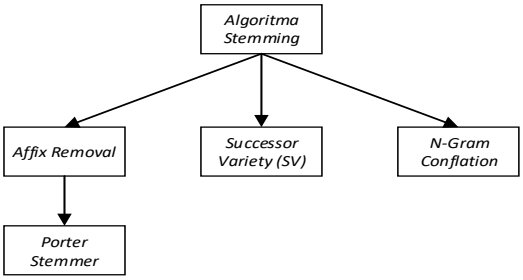

Figure 1. Diagram of Stemming

1. Affix Removal: removing suffixes and prefixes from a term to be its stem. The algorithm commonly used is the Porter algorithm because it has a simple model and is efficient.
   a. If a word ends with “ies” but not “eies” or “aies”, “ies” is replaced with “y”.
   b. If a word ends with “es” but not “aes” or “ees” or “oes”, “es” is replaced with “e”.
   c. If a word ends with “s” but not “us” or “ss”, “s” is replaced with “NULL”.
2. *Successor Variety (SV)*: emphasizes the structure of the letter in the word than the consideration on phonemes. The examples are: corpus, able, accident, ape, about, result in SV for the word “apple:
   a. Since the first letter of the word “apple” is “a”, so the word group that has substring “a” followed by “b”, “x”, “c”, “p” is called SV from “a” so “a” has 4 SV.
   b. Since the first two letters of the word “apple” are “ap” so the word group that has substring “ap” is only followed by “e” and called SV from “ap” so “ap” has 1 SV.
3. *N-Gram Conflation*: the basic idea of this technique is by grouping the words together based on the characters (substring) identified as N character.

Nazief and Adriani algorithm is a stemmer that was first developed by Bobby Nazief and Mirna Adriani. Nazief and Adriani algorithm is the best algorithm in the process of stemming because it has additional rules of reduplications, prefixes, and suffixes in improving the accuracy of each word. This algorithm is based on the comprehensive morphological rules of the Indonesian language, collected to be one group and encapsulated on allowed affixes and disallowed affixes. This algorithm uses the stem/ root word dictionary and supports recoding, which is restructuring the words that experienced the over-stemming process.[8]

The algorithm developed by Bobby Nazief and Mirna Adriani has the following stages [9]:

1. Find the word to be stemmed in the dictionary. If it is found, it can be assumed that the word is a stem or root word, so the algorithm stops.
2. The inflection of suffixes (“-lah”, “-kah”, “-ku”, “-mu”, or “-nya”) are removed. If it is a particle “-lah”, “-kah”, “-tah” or “-pun”), this stage is repeated to remove the possessive pronoun (“-ku”, “mu”, or “nya”), if it exists.
3. Remove suffix derivation (“-i”, “-an” or “-kan”). If the word is found in the dictionary, the algorithm stops. If it is not found, it goes to stage 3a.

   a. If “-an” has been removed and the last letter of the word is “-k”, so “-k” is also removed. If the word is found in the dictionary, the algorithm stops. If it is not found, it goes to stage 3b.

   b. The suffixes to be removed ("-i", "-an" or "-kan") is restored and continue to stage 4.
4. Remove prefix derivation. If in stage 3, a suffix is removed, it then goes to the stage 4a. If no, it goes to stage 4b.
   a. Check the table of prefix and suffix combination which is not allowed. If it is found, the algorithm stops, if not, it goes to the stage 4b.
   b. In this stage, the iteration is carried out three times. Decide the prefix type, and then remove the prefix. If the root word has not been found, go to stage 5. If it is found, the algorithm stops.

Note: if the second prefix is similar to the first prefix, the algorithm stops.

5. Recoding

6. If all stages have ended, but it still fails, the word is assumed as the root word, then the process ends.

The type of prefix is decided through these following steps:
1. If the prefixes are: "di-", "ke-", or "se-", the type of prefix is successively "di-", "ke-", or "se-".
2. If the prefixes are "te-", "me-", "be-", or "pe-", it needs an additional process to decide the type of the prefixes.
3. If the two first characters are not "di-","ke-","se-", "te-", "be-", "me-", or "pe-" so it stops.
4. The type of prefixes is none, so it stops.

The Javanese language is the language used by the society living in Central Java, East Java, and the Special Region of Yogyakarta. The Javanese language has several courtesies those are Ngoko Javanese (low), Madyo Javanese (middle), and Kromo Javanese (high refined). The use of the three courtesy is different; for example, Ngoko is used to communicate with peers and the elder to the younger, Kromo is usually used to communicate with the elder or respectful people, while Madya is the mix between Ngoko and Kromo. Affixed words in the Javanese language are divided into four types those are [10]:

a. *Prefixes.* Verbs added with prefixes before the root words. The prefixes could be n-, ma-, ka-, ke-, ko-, dak-, di-, pi-. Example:

Makarya (ma + karya "work") work
Digawa (di + gawa "bring") being brought

b. *Infixes.* Verbs inserted with infixes in the middle of the root words. The infixes could be –um, -in, -el, -er. Example:

Gumuyu (guyu "laugh" + -mu-) laugh
Tumindak (tindak "act" + -um-) act

c. *Suffixes*. Verbs added with suffixes after the root words. The suffixes could be –ake, -na, -ne, -ana, -ing, -mu. Example:

Jupukake (jupuk "take" + ake) take it
Gawakna (gawa "bring" + na) bring it

c. *Confixes*. Verbs added with prefixes before the root words and suffixes after them. The affixes could be di-ake, n-i, n-ake, in-an. Example:

Digawakake (gawa "bring" + di-ake) being brought
Njupuki (jupuk "take" + n-i) take
Several examples of affixed words in the Javanese language are in the following Table 1.

Table 1. The Examples of Affixed Words in Javanese and Indonesian Language

Javanese Root Word	Indonesian Root Word	Affixed Javanese Word	Affixed Indonesian Word
Gawa (Bring)	Bawa	Gawak***na***	Bawakan
Guyu (Laugh)	Tawa	G***um***uyu	Tertawa
Jupuk (Take)	Ambil	Jupuk***ake***	Ambilkan
Adoh (Far)	Jauh	***Ng***adoh	Menjauh
Itung (Calculate)	Hitung	***Ng***itung	Menghitung

2. Research Method

System flowchart is the figure of the stemming stages carried out by the system. Figure 2 shows the design of the system flowchart.

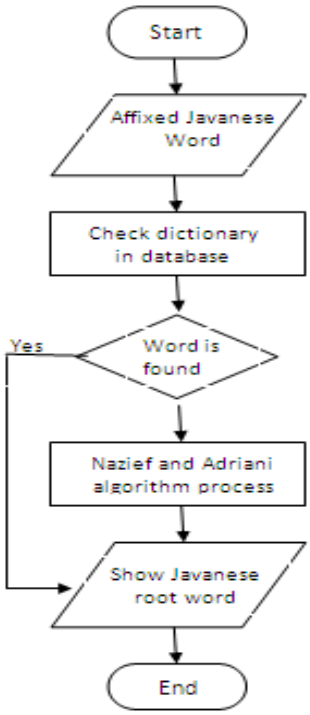

Figure 2. System Design

3. Result and Discussion

Data need analysis is the stage of deciding the limitation of data used. Data needed is data of affixes and root words in the Javanese language. Data grouping of Javanese affixed words of each category is shown in Table 2.

Table 2. Affixed Word Grouping

Category		Word Forming Affixes
inflectional suffixes	Particle	-a -ana -en
	inflectional suffixes possessive pronoun	-e -ku -mu -ne
Derivation Suffix		-ake -an -ke -na -ing -i
Derivation Prefix		Di- Dak- Kok- Nge- A- Ma- Ka- Ke- Sa-

	Pa- Pi- Pra- Pari- Kuma- Kami- Kapi- Tar- Nya- Ng- N- M-

A sample case of stemming Javanese based on the flowchart of Nazief and Adriani algorithm is shown in Figure 3.

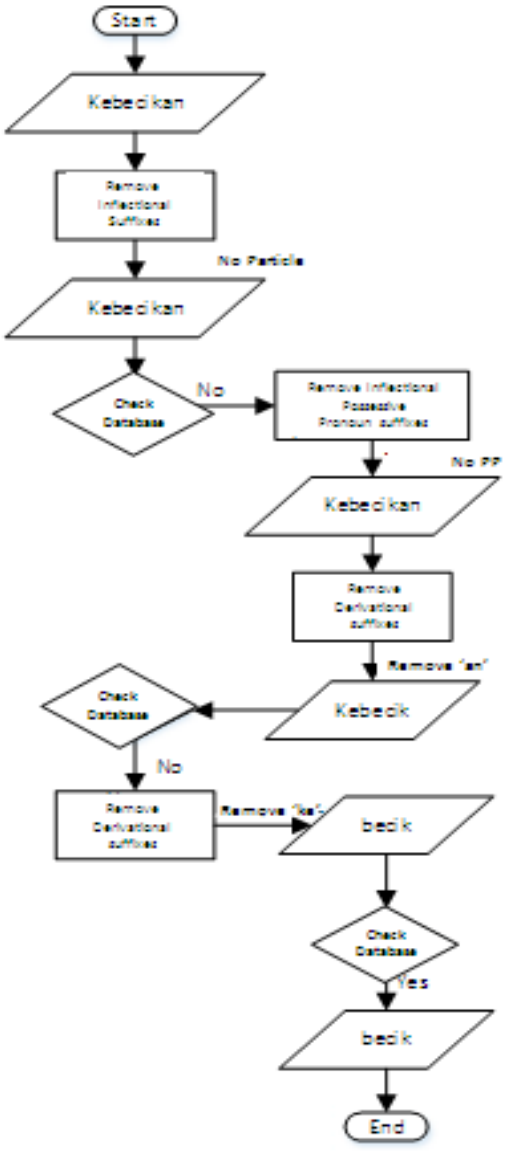

Figure 3. The Sample of Word Search Using Nazief and Adriani Algorithm

Nazief dan Adriani algorithm has 4 categories of affixes data grouping.

Algorithm and System *Pseudo-code* used in this research

a. Algorithm

Process 1: input the affixed Javanese word.

Process 2: check the root word dictionary on the database.

Process 3: the abstraction process of the root words and the affixes by using Nazief and Adriani algorithm.

Process 4: if the word is found in the database, the word inputted is assumed as the root word, and the stemming process will stop. However, if the word is not found, the system will go back to the input page.

b. *System Pseudo-code*

The following listing is Pseudo-code made based on the system flowchart as in Figure 4.

```
1.  Algorithm (Stemming to decide the root word from the
    affixes in Javanese)
2.  Declaration :
3.  Root word: string
4.  Result: string
5.  Description :
6.  Start
7.  Read (affixed Javanese 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 Algoritma Nazief dan Adriani"
    ("Run the process of Nazief and Adriani algorithm")
14. End.
```

Figure 4. Pseudo-code of system flowchart

The following Figure 5 is Pseudo-code made based on Nazief and Adriani algorithm flowchart.

```
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 (Javanese 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 5. Pseudo-code of Nazief and Adriani Algorithm flowchart

Database which used is root words and affixes in Javanese language as in Table 3.

Table 3. Root Words

Name	Type of Data	Length	Explanation
Id_katadasar	Int	10	Primary key
Katadasar (Root word)	Text	-	NULL
Tipe_katadasar	Text	-	NULL

The result of system implementation is as shown in the following Figure 6. The stemming result page shows the stemming process by using Nazief and Adriani algorithm and the root word as the result of the affixed Javanese word inputted.

Figure 6. Stemming Result Page

System testing is done to find out the capability of the system in finding the Javanese root words as in Table 4. This research used accuracy testing that is a test to measure the accuracy level of the system in finding the root words. Data used in this research was the affixed Javanese words.

Table 4. The Result of System Testing

	Affix	Affixed word	Root word	Root word on stem	Correct	Incorrect
Particles	-a	Maraa	Mara	Mara	✓	
		Sarapana	Sarapan	Sarap		✓
	-en	Panganen	Pangan	Pangan	✓	
		Gawanen	gawa	gawanen		✓
	-ana	Pakanana	Pakan	Pakan	✓	
Possesive Pronouns	-ku	Bukuku	Buku	Buku	✓	
		Panganku	Pangan	Pangan	✓	
	-mu	Ibumu	ibu	Ibu	✓	
		bapakmu	bapak	bapak	✓	
	-ne	Nyatane	Nyata	Nyata	✓	
		mejane	meja	meja	✓	
	-e	Pitike	Pitik	Pitik	✓	
		dhuwite	dhuwit	dhuwit	✓	
Derivasi	-ake	Jupukake	Jupuk	Jupuk	✓	

sufixes		Nulisake	Nulis	Nulis		
	-an	Gawean	Gawe	Gawe	✓	
		Takonan	Takon	Takon	✓	
	-i	Kemuli	Kemul	Kemul	✓	
		Mbalangi	Balang	Balang	✓	
	-ke	Jupukke	Jupuk	Jupuk	✓	
		Mbayarke	Bayar	Bayar	✓	
	-na	Gawakna	Gawa	Gawan		✓
		Golekna	Golek	Golek	✓	
	-ing	Pupusing	Pupus	Pupus	✓	
		Kawruhing	Kawruh	Kawruh	✓	
Prefixes	di-	Dipangan	Pangan	Pangan	✓	
		Dipikir	Pikir	Pikir	✓	
	Ke-	Kebecikan	Becik	Becik	✓	
		Kebutuhan	Butuh	Butuh	✓	
	Kok-	Kokgawe	Gawe	Gawe	✓	
		Kokpangan	Pangan	Pangan	✓	
	m-	Macul	Pacul	Pacul	✓	
		Mikul	Pikul	Pikul	✓	
	Dipun-	Dipunbukak	bukak	bukak	✓	
		dipunjagongi	jagong	jagong	✓	
	Ka-	Katabrak	Tabrak	Tabrak	✓	
		kapinteran	pinter	pinter	✓	
	Dak-	Dakpangan	Pangan	Pangan	✓	
		dakbukak	bukak	bukak	✓	
	Nge-	Ngecat	Cat	Cat	✓	
		ngepit	pit	Pit	✓	
		Prasekawan	Sekawan	Sekawan	✓	
		Pragen	Pragi	Prag		✓
	Tar-	Tarmulah	Mulah	Mulah	✓	
		tartantu	tantu	tantu	✓	
	Pari-	Parigawe	Gawe	Gawe	✓	
		Paripih	Pripih	Paripih		✓
	Kuma-	Kumakaruh	Karuh	Karuh	✓	
		kumayu	ayu	yu		✓
	Kami-	Kamigilan	Gila	Gilan		✓
		kamitenggengen	tenggeng	tenggeng	✓	
	Kapi-	Kapilare	Lare	Lare	✓	
		kapitunan	tuna	tunan		✓
	Ma-	magempuran	Gempur	Gempur	✓	
		manjel	anjel	Manjel	✓	
	Sa-	Sawengi	Wengi	Wengi	✓	
		salawang	lawang	lawang	✓	
	Pa-	Palilah	Lilah	Lilah	✓	
		Padhusunan	Dhusun	Dhusun	✓	
	a-	Apangan	Pangan	Pangan	✓	
		anganggo	anggo	anggo	✓	
	Pi-	Pitutur	Tutur	Tutur	✓	
		Piweling	Weling	Weling	✓	
	Ny-	Nyapu	Sapu	Sapu	✓	
		Nyunggi	Sunggi	Sunggi	✓	

Ng-	Ngombe nggoreng	Ombe goreng	Ombe goreng	✓
N-	Nambal	Tambal	Tambal	✓
	Nutup	Tutup	Tutup	✓

Accuracy Calculation:
Number of words: 219
Incorrect words: number of incorrect words $= \frac{number\ of\ incorrect\ words}{number\ of\ words\ tested}\ 100\%$

: number of incorrect words $= \frac{14}{219} x\ 100\%$ = 6.4%

Correct words: number of correct words $= \frac{number\ of\ correct\ words}{number\ of\ words\ tested} x\ 100\%$
: number of correct words $= \frac{205}{219} x\ 100\%$ = 93.6%

In this accuracy testing, several words could not be processed as in Table 5.

Table 5. Examples of Affixes that could not be processed by the system

Affix	Example	Explanation
Kami-	Kamigilan->gilan	In the word “kamigilan”, the output from the system is "gilan"; the system does not remove affix “n” because in the Javanese affixes there is no affix “n”, so the system does not remove it.
-na	Gawakna->Gawak	In the word “Gawakna” the output from the system is “gawak”, the system removes suffix “na”, but does not remove “k” because the system could not cut infix.

The percentage of the system accuracy is 93.6%, and 6.4% is the incorrect result.

4. Conclusion and Recommendation

This result of this research is a stemming application for affixed Javanese words that can be used to find root words of Ngoko Javanese affixed words by using Nazief and Adriani algorithm. The accuracy test that tested 219 words shows that the accuracy percentage of the system in finding root words is 93.6%.

Some recommendation for this research :

a. Affixes that still experienced incorrect processes, such as suffix "-na" and infixes could not be processed. Recommendation for further development is to use other methods to process the affixes that could not be processed in this research.
b. The assimilation of affixes at the end of the words could not be processed, such as "pa+turu+an" that becomes "paturon". The system could not find the root word.
c. It needs further research on prefixes and suffixes that are allowed and disallowed in the Javanese language.
d. This system can be developed with mobile technology to ease the users in accessing the system.

References

[1] Raharajo, S. H. (2010). *Kawruh Basa Jawa Pepak.* Semarang: Widya Karya.
[2] Simarangkir, M. S. (2017). STUDI PERBANDINGAN ALGORITMA - ALGORITMA STEMMING. *Jurnal Inkofar*, 40-46.

[3] Wibowo, J. (2016). Aplikasi Penentuan Kata Dasar dari Kata Berimbuhan pada Kalimat Bahasa Indonesia dengan Algoritma Stemming. *Jurnal Riset Komputer (JURIKOM)*, 346-350.
[4] Pramudita, H. R. (2011). PENERAPAN ALGORITMA STEMMING NAZIEF & ADRIANI DAN SIMILARITY . *Jurnal Ilmiah DASI* , 15-19.
[5] Asmara, D. A., Khairani, D., & Masruroh, S. U. (2013). Penerapan Algoritma Paice atau Husk untuk Stemming pada Kamus Bahasa Inggris ke Bahasa Indonesia. *Jurnal Teknik Informatika*.
[6] Amin, F. (2012). Sistem Temu kembali Informasi dengan metode Vector Space Model. *Fakultas Teknologi Informasi, Universitas Sitkubank*.
[7] Bunyamin, H. (2008). Aplikasi Information Retrieval(IR) CATA Dengan Metode Generalized Vector Space Model. *Jurnal Informatika* , 29-38.
[8] Wahyudi, D., Susyanto, T., & Nugroho, D. (n.d.). Implementasi dan Analisis Algoritma Stemming Nazief & Andriani dan Porter pada Dokumen Berbahasa Indonesia. *Jurnal Ilmiah SINUS*, 49-56.
[9] Nazief, B., & Adriani, M. (2007). Stemming Indonesian: A Confix-Stripping Approach. *ACM Transactions on Asian Language Information Processing* , 13-33.
[10] Wedhawati, & dkk. (2006). *Tata Bahasa Jawa Mutakhir*. Yogyakarta: Kanisius.