Journal of Information System Research (JOSH)
Volume 7, No. 2, Januari 2026, pp 249−259
ISSN 2686-228X (media online)
https://ejurnal.seminar-id.com/index.php/josh/
DOI 10.47065/josh.v7i2.7879

Comparative Analysis of Ahmad-Yusoff and Jaro-Winkler Approaches for Javanese Language Stemming

Aysza Belia Auly Andira, Fadhli Almu'iini Ahda*, Danang Arbian Sulistyo

Faculty of Technology and Design, Department of Informatics Engineering, Institut Teknologi dan Bisnis Asia, Malang
Jl. Soekarno Hatta - Rembuksari No. 1 A, Mojolangu, Lowokwaru District, Malang City, East Java, Indonesia
Email: [1]ayszabelia@gmail.com, [2*]adhi32286@gmail.com, [3]danangarbian@gmail.com
Correspondence Author Email: adhi32286@gmail.com
Submitted: **04/07/2025**; Accepted: **03/01/2026**; Published: **05/01/2026**

Abstract−This research presents a performance comparison between two approaches for identifying the base form of affixed Javanese words: the Ahmad Yusoff Sembok (AYS) rule-based stemming algorithm and the Jaro-Winkler (JW) string similarity approach. Javanese was selected as the focus because of its complex morphological structure, encompassing prefixes, suffixes, infixes, and confixes, along with significant speech-level and dialectal variation, which together pose challenges for natural language processing. The dataset comprises 720 manually annotated word lemma pairs. Evaluation was carried out using precision, recall, F1-score, accuracy, and Cohen's Kappa, complemented by error analysis on over-stemming and under-stemming cases. Results indicate that JW achieves higher overall performance (83.19% accuracy, 83% F1-score) compared to AYS (73.19% accuracy, 73% F1-score), with AYS producing more over-stemming errors (88 cases) and JW showing more under-stemming errors (47 cases). These outcomes suggest that similarity-based approaches are more effective in addressing Javanese morphological complexity, while also contributing a benchmark dataset of manually annotated Javanese word lemma pairs, a comparative evaluation framework between rule-based and similarity-based approaches, and practical insights for the development of stemming tools in regional languages that currently lack NLP resources.

Keywords: Stemming; Javanese Language; Ahmad Yusoff Sembok; Jaro-Winkler; Natural Language Processing; Overstemming; Understemming; Algorithm Evaluation

1. INTRODUCTION

Stemming is a fundamental step in natural language processing (NLP) that aims to reduce inflected or derived words to their root form. This process plays a vital role in minimizing morphological complexity, thereby improving the efficiency and accuracy of various NLP applications such as information retrieval, text classification, information extraction, and sentiment analysis[1], [2], [3], [4], [5]. For instance, different forms of a word like "running," "runner," and "ran" can be normalized to a common root, "run." Such normalization enhances the consistency of word representation in textual corpora and facilitates more effective linguistic analysis and computational processing. As NLP technology becomes increasingly widespread, numerous stemming algorithms have been developed, particularly for resource-rich languages. The Porter algorithm is widely adopted for English[6], while the Nazief & Adriani algorithm has become a standard for Indonesian[7], [8], [9], [10], [11]. These algorithms are typically tailored to the morphological characteristics of their respective languages and have shown strong performance in their native contexts. However, when applied to under-resourced regional languages such as Javanese which features complex morphological patterns and sociolinguistic variation the performance of these algorithms often declines significantly.

Javanese is one of the most widely spoken regional languages in Indonesia, predominantly used in Central Java, Yogyakarta, and East Java[12], [13], [14], [15], [16]. Despite its large speaker base, Javanese remains under-resourced in terms of digital linguistic tools and NLP infrastructure. Its rich and irregular morphological structure poses significant challenges. The language incorporates various types of affixes prefixes (ma-, dak-, ka-, ny-, ng-, kok-), infixes (-er-, -el-, -in-, -um-), suffixes (-ku, -mu, -an, -na, -ana, -en), and confixes (prefix-suffix combinations) and employs reduplication in word formation[17], [18], [19], [20], [21]. Furthermore, Javanese often exhibits non-linear morphological processes that cannot be addressed through simple truncation methods used in other languages like English or Indonesian[22], [23], [24], [25], [26]. In addition to its morphological complexity, Javanese also presents sociolinguistic challenges. It operates within a hierarchical speech level system consisting of ngoko (informal), madya (intermediate), and krama (formal), with vocabulary usage varying according to the speaker's social context and relationship[27]. These levels influence word forms and affixation patterns, further compounded by dialectal variations across regions[28], [29], [30], [31], [32]. Together, these factors create significant obstacles for applying general-purpose stemming algorithms to the Javanese language.

In light of these challenges, several studies provide a strong foundation for selecting suitable methods in addressing stemming for Javanese. Research such as Leveraging Jaro-Winkler for Enhanced Nazief-Adriani Banjar Text Stemming [17] has demonstrated the effectiveness of Jaro-Winkler similarity in improving stemming performance for another regional language, while A Comparative Study of Stemming Techniques on the Malay Text [18] emphasizes the importance of comparative analysis for linguistically related languages. Additionally, Perbandingan Levenshtein Distance dan Jaro-Winkler Distance untuk Koreksi Kata dalam Preprocessing Analisis Sentimen Pengguna Twitter [19] highlights the applicability of Jaro-Winkler in string similarity tasks, further justifying its relevance in NLP preprocessing. Meanwhile, Analisis dan Perbandingan Stemming Algoritma Porter

Journal of Information System Research (JOSH)
Volume 7, No. 2, Januari 2026, pp 249−259
ISSN 2686-228X (media online)
https://ejurnal.seminar-id.com/index.php/josh/
DOI 10.47065/josh.v7i2.7879

dengan Algoritma Ahmad Yusoff Sembok dalam Dokumen Teks Bahasa Indonesia [20] underscores the significance of the AYS algorithm in handling morphological variations in regional languages.

Building on these insights, this study focuses on two approaches: a rule-based method adapted from Ahmad Yusoff Sembok (AYS) and a string similarity-based method using Jaro-Winkler (JW). AYS offers flexibility in handling complex morphology without relying on large dictionaries, while JW provides a similarity-based strategy to address irregular word forms. These two methods were chosen not only for their adaptability but also because research applying them to regional languages particularly Javanese remains very limited. In contrast, the widely used Nazief & Adriani algorithm shows strong performance in Indonesian but struggles with non-standard affixes, dialectal differences, and hierarchical speech levels in Javanese, making AYS and JW more promising alternatives for this study.

To address these issues, the Ahmad Yusoff Sembok method was modified to suit Javanese morphology [2], while Jaro-Winkler compares affixed words to base forms using similarity scores [3]. Both methods were evaluated using precision, recall, F1-score, and Cohen's Kappa to assess accuracy and agreement with reference data [4]. The study contributes a comparative analysis of two methods for Javanese, offering insights into overstemming and understemming. While promising, limitations include a small dataset and unresolved morphological variations. Future work should expand the dataset and explore hybrid models to improve performance across regional languages.

2. RESEARCH METHODOLOGY

This study focuses on the application of two stemming algorithm approaches for the Javanese language, namely Ahmad Yusoff Sembok's rule-based algorithm and Jaro-Winkler's string similarity-based algorithm. These two algorithms are used to identify the base forms (lemmas) of inflected words in Javanese, representing two different paradigms of morphological processing: an explicit linguistic approach through affixation rules, and a statistical approach through character similarity measurements between words[1], [26].

The research process consisted of several main stages, starting with data collection, followed by preprocessing to prepare the data for processing by the system. Next, the processed data was tested using two separate stemming algorithms, namely the Ahmad Yusoff Sembok Stemmer and the Jaro-Winkler Stemmer. The stemming results from both methods are then evaluated in the algorithm testing stage using quantitative metrics such as accuracy, precision, recall, F1-score, as well as analysis of overstemming and understemming phenomena.

Each stage in the process is designed to ensure that the comparison is systematic and objective. In addition to quantitative evaluation, qualitative analysis is also conducted to identify the types of errors produced by each approach, especially in handling the complex morphological structure of the Javanese language. The sequence of this research process is illustrated in Figure 1, which presents the research flow chart.

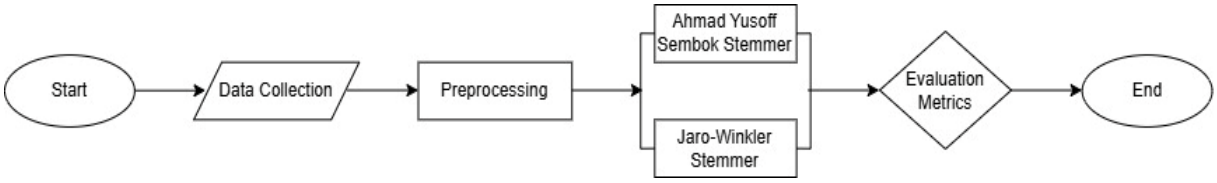

Figure 1. Research Flow Chart

Figure 1. Research Flow Chart illustrates the overall process of this study. The flow begins with data collection, where a set of affixed words and their root forms are gathered as the test dataset. The collected data then undergo preprocessing, including normalization and cleaning to ensure consistency. Subsequently, the data are processed using two stemming approaches, namely the Ahmad Yusoff Sembok Stemmer and the Jaro-Winkler Stemmer. Finally, the results are assessed using evaluation metrics such as precision, recall, F1-score, overstemming, and understemming, before the process reaches the end stage.

2.1 Data Collection

The data used in this study was obtained through the manual collection and compilation of a Javanese text corpus. The corpus comprises 720 annotated inflected words, including 314 words with prefixes, 61 with infixes, and 345 with suffixes. Each word is paired with its correct base form or lemma, enabling precise evaluation of the stemming algorithm's performance by comparing its output with the reference data. The data sources were drawn from a variety of references to ensure both diversity and representativeness. These include Javanese elementary school textbooks, which provide standard vocabulary in an educational context; the "Pepak" Javanese dictionary, which contains base word forms, affix types, and morphological rules; as well as various online sources such as blogs, social media platforms, forums, and Javanese-language news sites that contribute to vocabulary variation, including regional dialects and non-standard forms[40], [41], [42], [43], [44].

In total, the compiled dataset consisted of 720 annotated word–lemma pairs, comprising 314 words with prefixes, 61 with infixes, and 345 with suffixes. This distribution of affixed words is illustrated in Figure 2, which

Journal of Information System Research (JOSH)
Volume 7, No. 2, Januari 2026, pp 249−259
ISSN 2686-228X (media online)
https://ejurnal.seminar-id.com/index.php/josh/
DOI 10.47065/josh.v7i2.7879

shows the proportion of each affix type in the corpus. All data were manually annotated by native Javanese speakers with linguistic expertise to guarantee the accuracy of the pairing between affixed words and their corresponding lemmas. The annotated dataset was then stored in a plain text file (kata_uji.txt), which served as the input for testing the stemming algorithms.

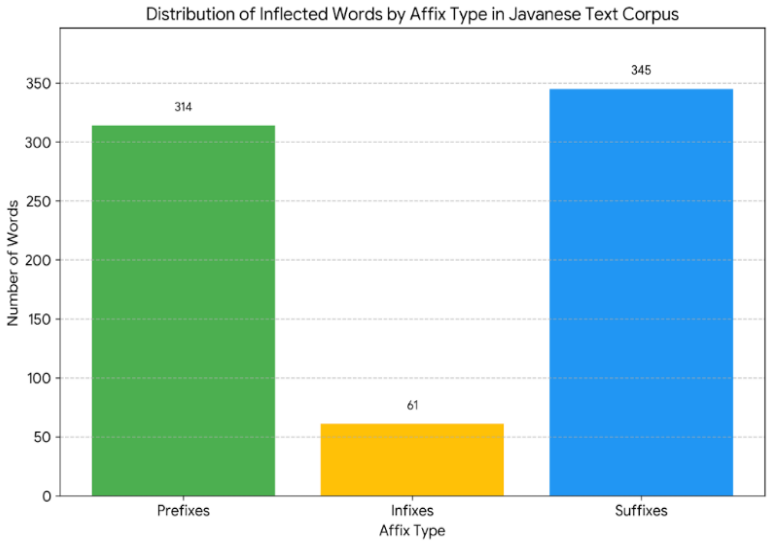

Figure 2. Affix Type Distribution in the Javanese Text Corpus

The structure and distribution of the data are illustrated in Figure 2. This figure shows the proportion of words with prefixes, infixes, and suffixes in the corpus, providing a clear overview of how different affix types are represented and ensuring that the dataset covers a balanced variety of morphological forms.

To provide a clearer overview of the data, Table 1 presents representative examples of the word lemma pairs along with their Indonesian and English meanings. These examples highlight how affixed forms are linked to their root words, thereby ensuring that the dataset is both systematic and practical for evaluating the performance of stemming algorithms.

Table 1. Dataset Overview

No	Word	Indonesian	English
1	anake	anak	child
2	anakan	anak	child
3	ambegan	bernafas	breathe
4	akrabake	akrab	familiar
5	adusake	memandikan	bathe
6	ajare	pengajaran	teaching
7	maguru	berguru	study
8	dijarah	dijajah	colonized
9	mbalang	melempar	throw
10	dibalang	dilempar	thrown
11	nggawa	membawa	lead
…	…	…	…
720	winoco	dibaca	read

As shown in Table 1, these representative word–lemma pairs illustrate how affixed words are systematically linked to their root forms, supporting both the quantitative distribution in Figure 2 and the practical evaluation of the stemming algorithms.

2.2 Preprocessing

The preprocessing stage is a critical initial step in the stemming system, as it transforms unstructured raw text into a clean and consistent format suitable for systematic analysis. In this study, preprocessing was conducted separately for the two algorithmic approaches employed, namely the Ahmad Yusoff Sembok (AYS) method and the Jaro-Winkler (JW) algorithm, with each step tailored to the specific characteristics of the respective method. One of the first processes in this stage involves reading the test data contained in kata_uji.txt, which includes pairs of affixed words and their corresponding base forms. The data are processed by splitting each line with a comma delimiter and storing it as a list of tuples, where each tuple consists of a test word and its associated root label. From these labels, a base word dictionary (kamus_jawa) is constructed to serve as a reference for validating

Journal of Information System Research (JOSH)
Volume 7, No. 2, Januari 2026, pp 249−259
ISSN 2686-228X (media online)
https://ejurnal.seminar-id.com/index.php/josh/
DOI 10.47065/josh.v7i2.7879

stemming results. In addition, lists of Javanese prefixes, suffixes, and infixes are defined to support the affix removal process. This procedure is outlined in the pseudocode provided in Table 2, which describes the workflow for reading test data, constructing the base word dictionary, and preparing the affix list.

Table 2. Pseudocode Algorithm

```
with open('kata_uji.txt', 'r', encoding='utf-8') as f:
    data_uji = []
    for line in f:
        parts = line.strip().split(',')
        if len(parts) == 2:
            data_uji.append((parts[0].strip(), parts[1].strip()))

kamus_jawa = {label for _, label in data_uji}
prefixes = ['ma', 'n', 'm', 'ny', 'ng', 'dak', 'kok', 'di', 'ka', 'sa', 'pa', 'pi', 'pra', 'pari', 'tar', 'kuma', 'kami', 'kapi']
suffixes = ['i', 'ku', 'mu', 'an', 'a', 'na', 'ana', 'en', 'ne', 'ing', 'ake', 'e']
infixes = ['in', 'um', 'em', 'el', 'er']
```

To illustrate the overall processing flow, Table 1 presents a pseudocode diagram showing the main steps in comparing AYS and JW. The process begins with loading the test data, followed by branching into two algorithmic paths, and concludes with evaluation and visualization of the stemming results. Each path applies its own preprocessing and stemming strategy before comparative performance measurement is conducted.

For the AYS approach, the system reads the data from kata_uji.txt and applies a sequential affix removal strategy. Prefixes (ma-, ny-, ng-, dak-, kok-, di-) are checked first; if the resulting form is found in the dictionary, the process stops and the word is accepted as the base form. If not, the algorithm proceeds with infix removal (-in-, -um-, -el-, -er-), followed by suffix removal (-ku, -mu, -an, -na, -ana, -en), and finally confixes (prefix–suffix combinations). Dictionary verification is performed after each step. This sequential rule hierarchy is summarized in Table 3, which illustrates representative examples of affix deletion. Such a process is intended to capture the characteristic morphological structure of the Javanese language. The results are then evaluated using precision, recall, F1-score, and Cohen's Kappa, alongside counts of over-stemming and under-stemming.

Table 3. Deletion process

No	Affixed Word	Process	Stemming	Method	Type	Results
1	maguru	ma - guru	guru	Ahmad Y Sembok	prefix	success
2	gumantung	g – um - antung	gantung	Jaro-Winkler	infix	success
3	tutupen	tutup - en	tutup	Ahmad Y Sembok	suffix	success
4	ditakoni	di – takon - i	takon	Jaro-Winkler	prefix + suffix	success

For the JW approach, preprocessing also begins with the same dataset, but the emphasis is on string similarity. Words first undergo light affix cleaning using the list of Javanese prefixes, suffixes, and infixes. Unlike AYS, which applies explicit rules and dictionary checks, JW compares the cleaned forms against the entire set of root words using the Jaro-Winkler similarity score. The candidate with the highest similarity score is then selected as the final stem. This step enables JW to identify root words even when affixation deviates from standard patterns, which often occurs in Javanese dialectal variations. The overall process of JW preprocessing and example outputs are also summarized in Table 3, which presents sample cases of affix handling and their stemming results. The JW results are evaluated using the same metrics as AYS, with additional analysis of error tendencies, including over-stemming and under-stemming.

In summary, AYS relies on a sequential rule-based affix removal strategy, while JW employs a similarity-based scoring approach with minimal affix cleaning. These distinct but complementary strategies are empirically compared in the evaluation stage, as illustrated in Figure 1 and further detailed in Table 3.

2.3 Evaluation Metrics

At this stage, the evaluation focused on assessing the performance of two stemming algorithms Ahmad Yusoff Sembok (AYS) and Jaro-Winkler (JW) in accurately identifying root words in Javanese. This step is essential for determining each algorithm's effectiveness in generating base forms that align with reference labels. To ensure objective measurement, several evaluation metrics were employed: Precision, Recall, F1-score, Cohen's Kappa, and Accuracy. Precision reflects the correctness of predicted stems, while Recall indicates the system's ability to retrieve all relevant root words. The F1-score balances these two aspects through their harmonic mean[34]. Cohen's Kappa evaluates the agreement between system output and reference labels, accounting for chance agreement. Accuracy provides an overall view of correct predictions relative to the total dataset. Together, these metrics offer a comprehensive assessment of the algorithms' performance. The equations used are presented below:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \times 100\% \quad (1)$$

Journal of Information System Research (JOSH)
Volume 7, No. 2, Januari 2026, pp 249−259
ISSN 2686-228X (media online)
https://ejurnal.seminar-id.com/index.php/josh/
DOI 10.47065/josh.v7i2.7879

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \times 100\% \quad (2)$$

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (3)$$

$$\text{Cohen's Kappa} = \frac{P_o - P_e}{1 - P_e} \quad (4)$$

$$\text{Accuracy} = \frac{\text{TP}}{\text{N}} \times 100\% \quad (5)$$

In addition, as part of the error analysis, the system also automatically calculates the number of overstemming and understemming cases that occur during the stemming process. Identifying these types of errors helps reveal patterns of weakness that the algorithm still has, especially in handling complex morphological variations. In general, the higher the values of Precision, Recall, F1-score, Cohen's Kappa, and Accuracy, the better the system's performance in generating base words that match the tested reference labels. Thus, the evaluation not only assesses technical accuracy but also reflects the algorithm's precision and reliability in addressing the linguistic challenges of the Javanese language.

3. RESULT AND DISCUSSION

3.1 Preprocessing

During the preprocessing stage, a series of data cleaning and transformation processes are carried out to prepare unstructured raw text into a neater format that is ready to be processed by the Ahmad Yusoff Sembok (AYS) and Jaro-Winkler (JW) algorithms. This stage aims to improve the accuracy of stemming results by ensuring that the data is free from disruptive elements, such as non-alphabetic characters, spelling errors, and inconsistencies in letter formatting.

Table 4. Preprocessing Results

No	Affixed Word	Affix	Root Word	Indonesian	English
1	anake	-e	anak	anak	child
2	anakan	-an	anak	anak	child
3	ambegan	-an	ambeg	bernafas	breathe
4	akrabake	-ake	akrab	akrab	familiar
5	adusake	-ake	adus	memandikan	bathe
6	ajare	-e	ajar	pengajaran	teaching
7	maguru	ma-	guru	berguru	study
8	dijarah	di-	jarah	dijajah	colonized
9	mbalang	m-	balang	melempar	throw
10	dibalang	di-	balang	dilempar	thrown
11	nggawa	ng-	gawa	membawa	lead
…	…	…	…	…	…
720	winoco	-in-	woco	dibaca	read

The preprocessing process includes removing affixes (prefixes, suffixes, and infixes), normalizing words, and simplifying word forms to match the algorithm's input format. The results of this stage are presented in Table 4, which shows how word forms are transformed before being entered into the stemming stage. For example, the word "anake" is simplified to "anak", and 'winoco' is changed to "woco." Such transformations are important to ensure that the analyzed data is clean, consistent, and aligned with the morphological structure targeted by the algorithm. Thus, the preprocessing stage serves as a crucial foundational step in supporting the overall performance of the stemming system.

3.2 Stemming Results

At this stage, words that have passed through the preprocessing stage are further processed through the stemming stage. This study uses two main approaches, namely the Ahmad Yusoff Sembok (AYS) algorithm and the Jaro-Winkler (JW) algorithm. The stemming process using the AYS approach is carried out in three stages, namely the removal of prefixes, infixes, and suffixes. The first step, prefix stemming, aims to remove common prefixes in Javanese. The second step, infix stemming, focuses on removing infixes inserted within words. Finally, suffix stemming is performed to remove suffixes attached to the end of words. This process is divided into three stages to improve accuracy in recognizing base forms, given the complex morphological structure of Javanese with affixation variations that affect meaning.

Meanwhile, the Jaro-Winkler approach is a string similarity-based method. This algorithm works by comparing the preprocessed words with all entries in the base word dictionary, then calculating the similarity score using the Jaro-Winkler metric. The base word with the highest similarity value that exceeds a certain threshold,

Journal of Information System Research (JOSH)
Volume 7, No. 2, Januari 2026, pp 249−259
ISSN 2686-228X (media online)
https://ejurnal.seminar-id.com/index.php/josh/
DOI 10.47065/josh.v7i2.7879

for example 0.85, will be selected as the stemming result. This approach is particularly useful for handling words with complex morphophonemic structures or infixes that are difficult to recognize explicitly through affixation rules.

By combining these two approaches, the stemming process becomes more comprehensive. The AYS algorithm offers advantages in systematically analyzing morphological structures, while the JW algorithm provides an alternative similarity-based solution for words that are not covered by explicit rules. The stemming results from both methods are then compared to evaluate their accuracy and effectiveness in identifying the base form of words in Javanese.

3.3.1 Prefix Stemming

The stemming process was performed on 314 Javanese words beginning with prefixes. For example, the word "maguru" was successfully reduced to the root form "guru," with the prefix "ma-" correctly recognized and removed. The complete results for prefix words are summarized in Table 5, which compares the performance of the Ahmad Yusoff Sembok (AYS) algorithm and the Jaro-Winkler algorithm.

Table 5. Results of Stemming Prefixes Using Ahmad Y Sembok and Jaro-Winkler Algorithms

Algorithm	Total Data	Success Rate	Overstemming	Understemming
Ahmad Y Sembok	314	284	11	19
Jaro-Winkler	314	268	26	20

Overall, both methods showed satisfactory performance with AYS successfully processing 284 out of 314 words and JW 268 out of 314 words, as summarized in Table 5. For example, the word "dijarah" was successfully converted to "jarah" and "ngombe" to "ombe" by both algorithms. This indicates that both the AYS and Jaro-Winkler algorithms are consistently capable of recognizing and removing various prefix forms in Javanese, such as (ma-, n-, m-, ny-, ng-, dak-, kok-, di-, and ka).

However, differences in results were still found in some words, particularly in cases with more complex morphological structures. These differences are typically related to the algorithm's sensitivity in identifying the root word from more varied forms. However, based on the data presented, most of the test words, including words such as "dibutuhake" which became "butuh" and "ditakoni" which became "takon", showed identical results. This finding indicates that both rule-based approaches such as AYS and string similarity-based approaches such as Jaro-Winkler are very effective in handling the prefix stemming process in Javanese.

3.3.2 Infix Stemming

The stemming process for infixes was performed on 61 words containing infixes in Javanese, such as (-in-, -um-, -l-, and -r-). Infixes are a type of affix inserted into the root word, which often affects the phonological and morphological structure of the word. A common example is the word "gumantung", which changes to "gantung" after the infix "-um-" is successfully removed through the stemming process. Handling infixes is important for improving the effectiveness of morphological processing in Javanese. The complete results for infixed words are summarized in Table 6, which compares the performance of the two algorithmic approaches.

Table 6. Results of Stemming Infixes Using Ahmad Y Sembok and Jaro-Winkler Algorithms

Algorithm	Total Data	Success Rate	Overstemming	Understemming
Ahmad Y Sembok	61	40	1	20
Jaro-Winkler	61	49	3	9

Both algorithms demonstrate reliable performance in identifying and removing infixes, as shown by the number of words successfully stemmed. As an illustration, the word "gumantung" was successfully changed to "gantung", "tinangis" to "tangis" and "sinerat" to "serat" by both methods. This indicates that both the AYS and Jaro-Winkler approaches are capable of handling various types of insertions such as (-um-, -in-) and complex patterns such as (-in-...-a) with a reasonable level of accuracy. However, there are still some words that failed to undergo infix removal, such as "mlempem", "mlebu", and "jlerit." In these cases, neither the rule-based algorithm nor the string similarity algorithm could identify the hidden or morphologically irregular infixes, resulting in under-stemming cases. This finding indicates that further refinement is needed in both approaches to make them more adaptive to unconventional insertion patterns. Nevertheless, the results in Table 6 show that both methods exhibit promising capabilities in infix stemming for Javanese.

3.3.3 Suffix Stemming

The stemming process for words containing suffixes in Javanese was performed on 345 words. One example that was successfully processed was the word "anake", which was returned to its base form "anak" by removing the suffix "-e". This demonstrates the algorithm's ability to accurately detect and separate suffixes from the root word. The complete results are summarized in Table 7, which presents the total data, success rate, and the number of over-stemming and under-stemming cases for both approaches. According to the table, AYS successfully

Journal of Information System Research (JOSH)
Volume 7, No. 2, Januari 2026, pp 249−259
ISSN 2686-228X (media online)
https://ejurnal.seminar-id.com/index.php/josh/
DOI 10.47065/josh.v7i2.7879

processed 203 words, with 76 over-stemming and 66 under-stemming cases, while JW successfully processed 282 words, with 18 over-stemming and 45 under-stemming cases.

Table 7. Results of Stemming Suffixes Using Ahmad Y Sembok and Jaro-Winkler Algorithms

Algorithm	Total Data	Success Rate	Overstemming	Understemming
Ahmad Y Sembok	345	203	76	66
Jaro-Winkler	345	282	18	45

From these results, as summarized in Table 7, it is evident that suffix removal generally proceeds smoothly. For example, the word "adusake" was successfully converted to "adus" and "gambaran" to "gambar". Both algorithms tested by Ahmad Yusoff Sembok and Jaro-Winkler were able to eliminate various types of suffixes such as -e, -an, -ake, and similar forms without obscuring the meaning of the root word. However, some differences between the methods are notable. For example, for the word "panganan", the Ahmad Yusoff Sembok algorithm produces the form "ngan", indicating over-stemming by removing part of the root word "pangan." whereas JW retained "panganan", indicating under-stemming due to the failure to detect and remove the suffix. This difference highlights the characteristics of each method: rule-based algorithms tend to be more aggressive in truncation, while string similarity-based approaches are more conservative and cautious.

Overall, both algorithms demonstrate fairly stable performance in handling suffixed words. Nonetheless, further refinement is needed to improve the identification of basic morpheme boundaries, balancing cutting effectiveness with semantic accuracy, thereby minimizing potential over-stemming and under-stemming and making the Javanese suffix stemming process more optimal and contextually appropriate.

3.4 Qualitative Error Analysis

To complement the quantitative results, a qualitative analysis was conducted by analyzing representative cases of over-stemming and under-stemming for each type of affix. For prefixes, a case of over-stemming occurred in the word "kapilih" which was stemmed to "lih", when the correct base form is "pilih". This shows that the AYS rule is too aggressive and cuts not only the prefix "ka-"but also part of the base word. Meanwhile, JW produced under-stemming in the word "ateges", which remained "ateges", whereas the correct base form is "teges". This occurred because the similarity score failed to completely remove the prefix a-. For infixes, AYS showed over-stemming in the word "kumawani", which was produced as wan, while the correct base form is "mawan". This error occurs because the infix -um- is processed incorrectly, causing an essential part of the root word to be deleted. Conversely, JW produces under-stemming on the word "akudhung", which is still produced as "akudhung", whereas the correct root form is "kudhung". This shows that affixes are not fully detected during the similarity matching process. For suffixes, AYS produces over-stemming on the word "panganan", which is produced as "ngan", whereas the correct root form is "pangan". This error is caused by excessive truncation that goes beyond the suffix -an. On the other hand, JW produces under-stemming on the word "diwutah", which is produced as "wutah", whereas the correct base form is "Utah". This is caused by a threshold mechanism that fails to completely remove the initial affix, leaving elements in the stemming result. These representative cases show that AYS tends to over-stem due to the application of rigid rules, especially when affixes resemble root segments, while JW is more prone to under-stemming due to its similarity-based approach, which is often more conservative, thus retaining partially affixed word forms when phonological variations occur in the root word.

3.5 Evaluation Metrics

The testing was conducted by calculating evaluation metrics such as Accuracy, Precision, Recall, F1-score, and Cohen's Kappa. The stemming process was performed thoroughly on words in Javanese. Both algorithms successfully found the root words, but there were differences in the number of over-stemming and under-stemming. A comparison of the test results of the two methods is shown in Table 8.

Table 8. Comparison Results: Ahmad Y Sembok vs. Jaro-Winkler Stemming

Metric	Ahmad Yusoff Sembok	Jaro-Winkler
Amount of Test Data	720	720
Correct Amount	527	599
Over-stemming	88	47
Under-stemming	105	74
Accuracy (%)	73.19%	83.19%
Precision (%)	73%	83%
Recall (%)	73%	83%
F1-Score (%)	73%	83%
Cohen's Kappa	73%	83%

As shown in Table 8, the Jaro-Winkler algorithm consistently outperformed the Ahmad Yusoff Sembok method, achieving high accuracy (83.19% vs. 73.19%) and producing fewer over-stemming and under-stemming

Journal of Information System Research (JOSH)
Volume 7, No. 2, Januari 2026, pp 249−259
ISSN 2686-228X (media online)
https://ejurnal.seminar-id.com/index.php/josh/
DOI 10.47065/josh.v7i2.7879

errors. This highlights JW's more precise handling of Javanese morphology, while AYS remains prone to excessive affix removal.

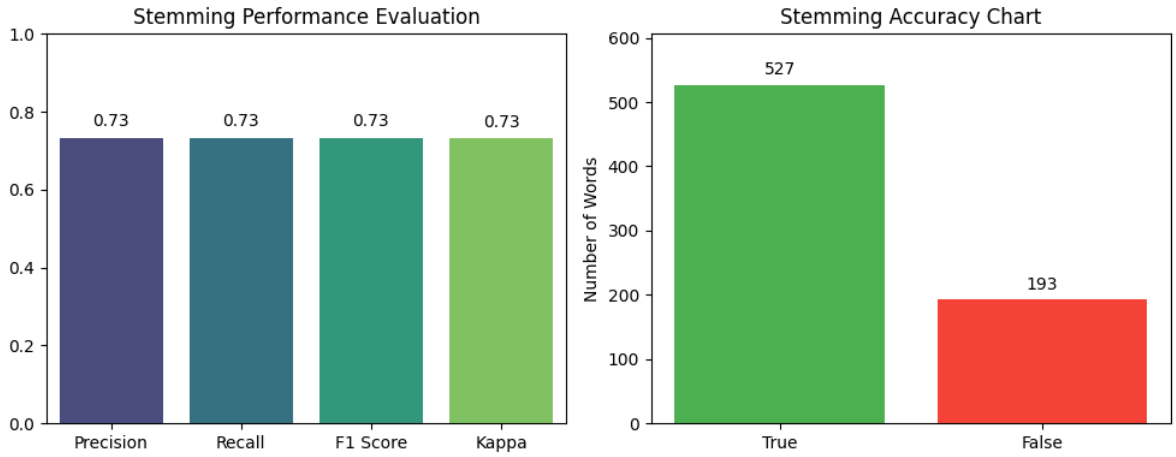

Figure 3. Ahmad Y Sembok Algorithm Bar Chart

Figure 3 illustrates the performance evaluation of the AYS stemmer. The left chart shows that Precision, Recall, F1-Score, and Cohen's Kappa all reached 0.73, indicating a balanced but moderate level of performance. The right chart displays the stemming accuracy, where 527 words were correctly stemmed (True) and 193 words were incorrectly stemmed (False). These results confirm that although the AYS method can process a large portion of the dataset successfully, it still leaves room for improvement, particularly in reducing error cases.

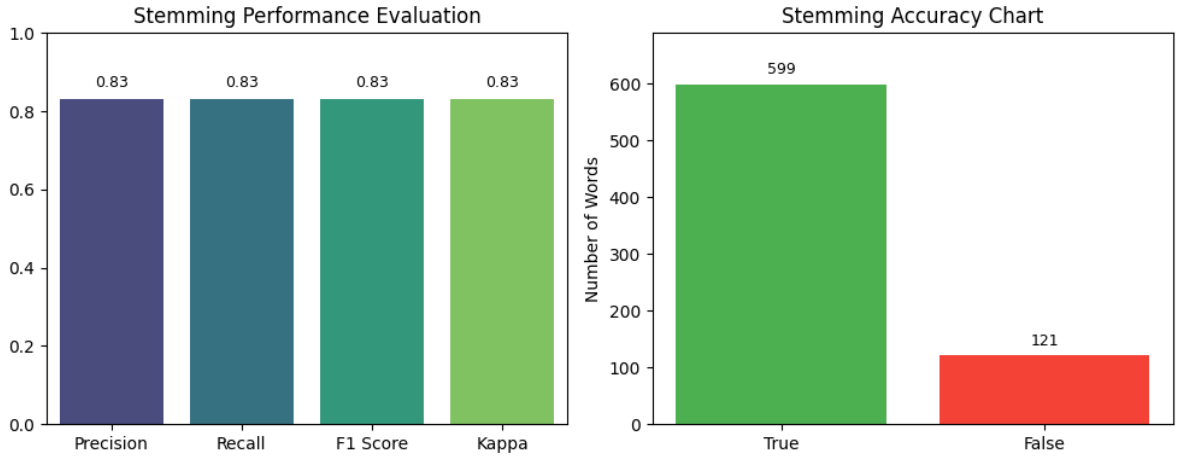

Figure 4. Jaro-Winkler Algorithm Bar Chart

Figure 4 shows that the Jaro-Winkler algorithm achieved consistent Precision, Recall, F1-Score, and Cohen's Kappa scores of 0.83. The accuracy graph on the right shows that 599 words were correctly classified and 121 words were misclassified. These results confirm the superiority of JW in providing higher accuracy compared to AYS.

Table 8 shows the comparative performance of the Ahmad Yusoff Sembok and Jaro-Winkler algorithms in stemming Javanese words. Out of 720 test words, AYS achieved an accuracy of 73.19% with 527 correct results, but it produced more errors with 88 cases of over-stemming and 105 cases of under-stemming. In contrast, JW reached a higher accuracy of 83.19% with 599 correct results, and fewer errors overall, recording only 47 cases of over-stemming and 74 cases of under-stemming. These results, consistent with Figure 3 and Figure 4, confirm that JW is more accurate and reliable than AYS in handling Javanese morphology.

3.5 Discussion

In previous studies, stemming algorithms have been widely used to process regional languages, including Javanese. One of the most commonly applied approaches is a modification of rule-based algorithms tailored to the morphological structure and affixation patterns of Javanese, such as adaptations of the Nazief & Adriani algorithm. In this study, we applied the Ahmad Yusoff Sembok (AYS) and Jaro-Winkler (JW) algorithms to 720 Javanese test words. The results indicate that both algorithms are capable of identifying root words with prefixes, suffixes, or combinations of both; however, their performance differs significantly. AYS achieved an accuracy of 73.19%, with 88 cases of over-stemming and 105 cases of under-stemming, showing that affix processing is not yet fully

Journal of Information System Research (JOSH)
Volume 7, No. 2, Januari 2026, pp 249−259
ISSN 2686-228X (media online)
https://ejurnal.seminar-id.com/index.php/josh/
DOI 10.47065/josh.v7i2.7879

optimal and tends to remove parts of words excessively. By contrast, JW achieved a higher accuracy of 83.19%, with 47 over-stemming and 74 under-stemming cases, demonstrating a more careful approach that preserves root forms. These tendencies are clearly visualized in Figure 5, which compares the number of over-stemming and under-stemming cases between the two methods.

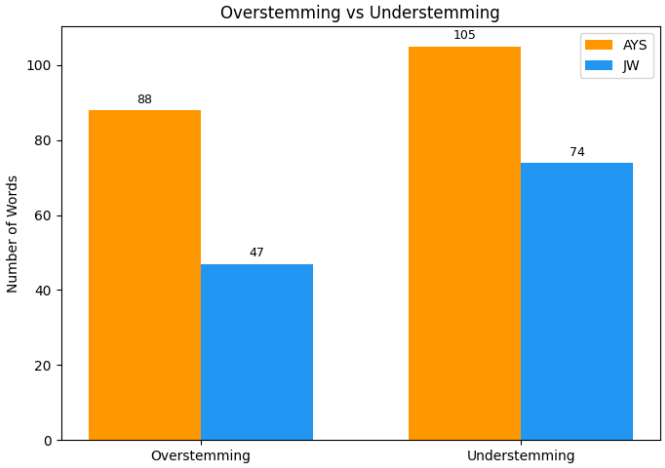

Figure 5. Overstemming vs Understemming in Ahmad Y Sembok and Jaro-Winkler Methods

Figure 5 further illustrates that AYS frequently overstems because its affix removal rules can be too aggressive, for example reducing “panganan” to “ngan” or “kancaana” to “nca”. JW, on the other hand, produces fewer overstemming cases because it relies on similarity scoring rather than strict rules, though it still generates some understemming, such as “tujuwane” becoming “tujuwa”. These patterns confirm the strengths and limitations of each approach: AYS tends to over-truncate when suffixes resemble root segments, while JW is more conservative but can fail when phonological variation is too large.

Compared to other rule-based adaptations that also struggled with overstemming in previous studies, these findings are quite consistent, while the similarity-based approach tends to offer higher precision. The results of this study reinforce the idea that string similarity methods are more robust for agglutinative languages such as Javanese, where morphological variation is irregular. In practice, JW is more suitable for applications that require high precision and minimal overstemming, such as search engines or digital libraries. Meanwhile, AYS can still be valuable in contexts that prioritize broader coverage, albeit with a higher error rate. A promising direction for development is the creation of hybrid models that combine linguistic rules with similarity-based mechanisms, thereby balancing accuracy and flexibility in Javanese stemming.

4. CONCLUSION

Based on the results of stemming research on Javanese using the Ahmad Yusoff Sembok (AYS) and Jaro-Winkler (JW) algorithms, several conclusions can be drawn. The AYS algorithm achieved an accuracy of 73.61%, with 95 cases of over-stemming and 31 cases of under-stemming, while the JW algorithm demonstrated better performance with an accuracy of 82.84%, only 7 cases of over-stemming, and 44 cases of under-stemming. Error analysis revealed that AYS tends to over-stem due to overly aggressive affix removal, whereas JW is more conservative but results in under-stemming because some affixes remain unremoved. Despite these differences, both methods showed relatively high effectiveness in handling the morphological structure of the Javanese language. To further improve performance, adjustments to affix rules including special handling of morphemes, infixes, confixes, and repetitions are needed, along with expansion of the dataset to cover dialectal variations, speech levels, and phrase forms. Future development may also consider combining rule-based and similarity-based approaches in a hybrid model, integrating machine learning to recognize complex morphological patterns, optimizing the JW similarity threshold, and applying the stemming system in real-world NLP tasks such as information retrieval or sentiment analysis to enhance its practicality and adaptability.

REFERENCES

[1] W. D. Suryono, E. Utami, and D. Ariatmanto, “Analisa Perbandingan Stemming Dokumen Teks Berbahasa Jawa dengan Algoritma Levenshtein Distance Dan Jaro-Winkler,” JIPI J. Ilm. Penelit. Dan Pembelajaran Inform., vol. 10, no. 1, pp. 774–780, Jan. 2025, doi: 10.29100/jipi.v10i1.6092.

[2] D. A. Sulistyo, A. P. Wibawa, D. D. Prasetya, and F. A. Ahda, “An enhanced pivot-based neural machine translation for low-resource languages,” Int. J. Adv. Intell. Inform., vol. 11, no. 2, Art. no. 2, May 2025, doi: 10.26555/ijain.v11i2.2115.

[3] A. R. R. Ivani, A. Z. Kurniadi, and A. B. A. Andira, “VGG16 untuk Klasifikasi Tingkat Kematangan pada Buah Apel di Kota Batu,” 2025.

Journal of Information System Research (JOSH)
Volume 7, No. 2, Januari 2026, pp 249−259
ISSN 2686-228X (media online)
https://ejurnal.seminar-id.com/index.php/josh/
DOI 10.47065/josh.v7i2.7879

[4] M. Syahrullah, F. H. Rachman, and I. O. Suzanti, "Deteksi Kemiripan Dokumen Abstrak Skripsi menggunakan Metode Jaro-Winkler Distance dan Synonym Recognition," Sains Data J. Studi Mat. Dan Teknol., vol. 2, no. 2, pp. 68–79, Dec. 2024, doi: 10.52620/sainsdata.v2i2.136.
[5] Muharir, E. Noersasongko, Muljono, A. Syukur, and M. Aqqad, "Leveraging Jaro-Winkler for Enhanced Nazief-Adriani Banjar Text Stemming," in 2024 Ninth International Conference on Informatics and Computing (ICIC), Oct. 2024, pp. 1–6. doi: 10.1109/ICIC64337.2024.10956530.
[6] A. Arif siswandi, Y. Permana, and A. Emarilis, "Stemming Analysis Indonesian Language News Text with Porter Algorithm," J. Phys. Conf. Ser., vol. 1845, no. 1, p. 012019, Mar. 2021, doi: 10.1088/1742-6596/1845/1/012019.
[7] M. Ashari, D. A. Sulistyo, and F. A. Ahda, "STEMMING IN MADURESE LANGUAGE USING NAZIEF AND ADRIANI ALGORITHM," J. Tek. Inform. Jutif, vol. 5, no. 4, Art. no. 4, July 2024, doi: 10.52436/1.jutif.2024.5.4.2012.
[8] F. A. Ahda, A. P. Wibawa, D. D. Prasetya, and D. A. Sulistyo, "Comparison of Adam Optimization and RMS prop in Minangkabau-Indonesian Bidirectional Translation with Neural Machine Translation," JOIV Int. J. Inform. Vis., vol. 8, no. 1, pp. 231–238, Mar. 2024, doi: 10.62527/joiv.8.1.1818.
[9] I. Afanasev and O. Lyashevskaya, "Chapter 2 String Similarity Measures for Evaluating the Lemmatisation in Old Church Slavonic," Brill, 2024. doi: 10.1163/9789004702660_003.
[10] Z. Abidin, A. Junaidi, and Wamiliana, "Text Stemming and Lemmatization of Regional Languages in Indonesia: A Systematic Literature Review," J. Inf. Syst. Eng. Bus. Intell., vol. 10, no. 2, pp. 217–231, June 2024, doi: 10.20473/jisebi.10.2.217-231.
[11] D. A. Sulistyo, A. P. Wibawa, D. D. Prasetya, and F. A. Ahda, "LSTM-Based Machine Translation for Madurese-Indonesian," J. Appl. Data Sci., vol. 4, no. 3, Art. no. 3, Sept. 2023, doi: 10.47738/jads.v4i3.113.
[12] Rina, "Memahami Confusion Matrix: Accuracy, Precision, Recall, Specificity, dan F1-Score untuk Evaluasi…," Medium. Accessed: June 23, 2025. [Online]. Available: https://esairina.medium.com/memahami-confusion-matrix-accuracy-precision-recall-specificity-dan-f1-score-610d4f0db7cf
[13] R. Mohemad, N. N. Mohd Muhait, N. M. Mohamad Noor, and N. F. Akma Mamat, "A Comparative Study of Stemming Techniques on the Malay Text.," Int. J. Adv. Comput. Sci. Appl., vol. 14, no. 12, p. 133, Dec. 2023, doi: 10.14569/ijacsa.2023.0141213.
[14] Kulampah, "Kamus Bahasa Jawa Dan Artinya [Lengkap Dan Update 2024]." Accessed: June 16, 2025. [Online]. Available: https://kulampah.com/kamus-bahasa-jawa-dan-artinya/
[15] H. Jayadianti, B. Santosa, J. Cahyaning, S. Saifullah, and R. Drezewski, "Essay auto-scoring using N-Gram and Jaro Winkler based Indonesian Typos," MATRIK J. Manaj. Tek. Inform. Dan Rekayasa Komput., vol. 22, no. 2, pp. 325–338, Mar. 2023, doi: 10.30812/matrik.v22i2.2473.
[16] V. F. Sopacua, R. A. Da Costa, and L. F. Pesiwarissa, "REDUPLIKASI DALAM BAHASA MELAYU AMBON (KAJIAN MORFOLOGI)," ARBITRER J. Pendidik. Bhs. Dan Sastra Indones., vol. 4, no. 2, pp. 687–704, Aug. 2022, doi: 10.30598/arbitrervol4no2hlm687-704.
[17] D. Kastowo, A. Saputra, W. D. Suryono, and E. Setyowati, "Analisis Perbandingan Algoritma Nazief Adriani dan Levenshtein Distance untuk mengukur Tingkat Similaritas Berita Menggunakan Rabin Krap: Studi Kasus Berita Berbahasa Jawa," JNANALOKA, pp. 1–10, Mar. 2022, doi: 10.36802/jnanaloka.2022.v3-no1-1-10.
[18] T. Efriyanto and M. Hayaty, "JARO WINKLER ALGORITHM FOR MEASURING SIMILARITY ONLINE NEWS," J. Tek. Inform. Jutif, vol. 3, no. 4, pp. 975–982, Aug. 2022, doi: 10.20884/1.jutif.2022.3.4.152.
[19] A. F. Aji et al., "One Country, 700+ Languages: NLP Challenges for Underrepresented Languages and Dialects in Indonesia," Mar. 24, 2022, arXiv: arXiv:2203.13357. doi: 10.48550/arXiv.2203.13357.
[20] M. A. Yulianto and N. Nurhasanah, "The Hybrid of Jaro-Winkler and Rabin-Karp Algorithm in Detecting Indonesian Text Similarity," J. Online Inform., vol. 6, no. 1, pp. 88–95, June 2021, doi: 10.15575/join.v6i1.640.
[21] A. P. Wibawa and M. N. Hakim, "STEMMING BAHASA JAWA MENGGUNAKAN DAMERAU LEVENSHTEIN DISTANCE (DLD)," J. Tek. Inform., vol. 14, no. 1, pp. 22–27, Sept. 2021, doi: 10.15408/jti.v14i1.15010.
[22] M. F. Tanjung, "Boosting Stemmer Performance Using Cache Method," J. Mat. Dan Ilmu Pengetah. Alam LLDikti Wil. 1 JUMPA, vol. 1, no. 1, pp. 6–9, Mar. 2021, doi: 10.54076/jumpa.v1i1.34.
[23] D. Sulistyo, F. Ahda, and V. A. Fitria, "Epistomologi dalam Natural Language Processing," J. Inov. Teknol. Dan Edukasi Tek., vol. 1, no. 9, pp. 652–664, Sept. 2021, doi: 10.17977/um068v1i92021p652-664.
[24] O. V. Putra, A. Musthafa, and K. P. Wibowo, "Klasifikasi Ekspresi Teks Berbahasa Jawa Menggunakan Algoritma Long Short Term Memory," Komputika J. Sist. Komput., vol. 10, no. 2, pp. 137–143, Aug. 2021, doi: 10.34010/komputika.v11i1.4616.
[25] M. A. Nur, "Perbandingan Levenshtein Distance Dan Jaro-Winkler Distance Untuk Koreksi Kata Dalam Preprocessing Analisis Sentimen Pengguna Twitter," J. Fokus Elektroda Energi List. Telekomun. Komput. Elektron. Dan Kendali, vol. 6, no. 2, p. 88, June 2021, doi: 10.33772/jfe.v6i2.17751.
[26] I. Huda, "IMPLEMENTASI NATURAL LANGUAGE PROCESSING (NLP) UNTUK APLIKASI PENCARIAN LOKASI," J. Nas. Teknol. Terap. JNTT, vol. 3, no. 2, Art. no. 2, Oct. 2021, doi: 10.22146/jntt.35036.
[27] R. Hayati, "VARIASI BAHASA DAN KELAS SOSIAL," Pena J. Ilmu Pengetah. Dan Teknol., vol. 35, no. 1, p. 48, Mar. 2021, doi: 10.31941/jurnalpena.v35i1.1348.
[28] A. Zaremba and E. Demir, "ChatGPT: Unlocking the future of NLP in finance," Mod. Finance, vol. 1, no. 1, pp. 93–98, 2023.
[29] D. Kiela et al., "Dynabench: Rethinking Benchmarking in NLP," Apr. 07, 2021, arXiv: arXiv:2104.14337. doi: 10.48550/arXiv.2104.14337.
[30] J. M. S. Efani, "NORMALISASI KATA BAHASA JAWA PADA TWEET DENGAN EDIT DISTANCE DAN DICTIONARY LOOKUP," PhD Thesis, UNIVERSITAS ISLAM NEGERI SULTAN SYARIF KASIM RIAU, 2021. Accessed: Sept. 28, 2025. [Online]. Available: https://repository.uin-suska.ac.id/50618/2/T.A%20JELITA.pdf
[31] W. D. Suryono, E. Utami, and D. Ariatmanto, "Analisa Perbandingan Stemming Dokumen Teks Berbahasa Jawa dengan Algoritma Levenshtein Distance Dan Jaro-Winkler," JIPI J. Ilm. Penelit. Dan Pembelajaran Inform., vol. 10, no. 1, pp. 774–780, 2025.

Journal of Information System Research (JOSH)
Volume 7, No. 2, Januari 2026, pp 249−259
ISSN 2686-228X (media online)
https://ejurnal.seminar-id.com/index.php/josh/
DOI 10.47065/josh.v7i2.7879

[32] - Novi Yulianti, "ALGORITMA STEMMING BAHASA WOLIO BERBASIS ATURAN MORFOLOGI," skripsi, Universitas Islam Negeri Sultan Syarif kasim Riau, 2021. Accessed: Sept. 28, 2025. [Online]. Available: https://repository.uin-suska.ac.id/54523/

[33] - Nur Hasanah Hrp, = Muhammad Fikry, and - Yusra, "ALGORITMA STEMMING TEKS BAHASA BATAK ANGKOLA BERBASIS ATURAN TATA BAHASA," ALGORITMA STEMMING TEKS Bhs. BATAK ANGKOLA Berbas. ATURAN TATA Bhs., vol. 4, no. 3, pp. 643–648, May 2023.

[34] Z. Abidin, A. Wijaya, and D. Pasha, "Aplikasi Stemming Kata Bahasa Lampung Dialek Api Menggunakan Pendekatan Brute-Force dan Pemograman C," J. Media Inform. Budidarma, vol. 5, no. 1, pp. 1–8, 2021.

[35] D. Mustikasari, I. Widaningrum, R. Arifin, and W. H. E. Putri, "Comparison of Effectiveness of Stemming Algorithms in Indonesian Documents," presented at the 2nd Borobudur International Symposium on Science and Technology (BIS-STE 2020), Atlantis Press, Aug. 2021, pp. 154–158. doi: 10.2991/aer.k.210810.025.

[36] D. O. Dewi, D. Oktafiani, and Y. Astica, "OPTIMASI ALGORITMA STEMMING PORTER UNTUK PEMROSESAN TEKS DALAM BAHASA INDONESIA: OPTIMASI ALGORITMA STEMMING PORTER UNTUK PEMROSESAN TEKS DALAM BAHASA INDONESIA," J. Inform. Dan Sist. Inf., vol. 6, no. 1, pp. 42–52, June 2025.

[37] M. U. Albab, Y. K. P, and M. N. Fawaiq, "Optimization of the Stemming Technique on Text Preprocessing President 3 Periods Topic," J. Transform., vol. 20, no. 2, pp. 1–12, Jan. 2023, doi: 10.26623/transformatika.v20i2.5374.

[38] L. Pertiwi, "Penerapan Algoritma Text Mining, Steaming Dan Texrank Dalam Peringkasan Bahasa Inggris," BIMASATI Bull. Multi-Discip. Sci. Appl. Technol., vol. 1, no. 3, pp. 100–104, 2022.

[39] W. E. S. Nurlina et al., Kamus bahasa Jawa-Indonesia. Yogyakarta: Balai Bahasa Provinsi Daerah Istimewa Yogyakarta, 2021. Accessed: Sept. 28, 2025. [Online]. Available: https://repositori.kemendikdasmen.go.id/28642/

[40] F. Nuryantiningsih, "Relevansi Adjektiva Human Propensity dalam Bahasa Jawa sebagai Cerminan Pandangan Hidup Manusia Jawa," Deskripsi Bhs., vol. 5, no. 2, pp. 50–57, Oct. 2022, doi: 10.22146/db.v5i2.5849.

[41] "Analisis Interferensi Morfologi Bahasa Jawa ke Bahasa Indonesia dalam Film â€œSepatu Dahlanâ€ Karya Benni Setiawan | Karim | Jurnal Bahasa, Sastra, dan Budaya." Accessed: Sept. 28, 2025. [Online]. Available: https://ejurnal.ung.ac.id/index.php/JBSP/article/view/16002

[42] "INTERFERENSI MORFOLOGIS BAHASA JAWA DALAM PENGGUNAAN BAHASA INDONESIA SISWA SMP IT NURUL IKHWAH NAGAN RAYA ACEH | Prosiding Konferensi Linguistik Tahunan Atma Jaya (KOLITA)." Accessed: Sept. 28, 2025. [Online]. Available: https://ejournal.atmajaya.ac.id/index.php/kolita/article/view/3779

[43] R. A. Sholihah, "VARIASI MORFOLOGI BAHASA JAWA PONOROGO: AFIKSASI, REDUPLIKASI, DAN PEMAJEMUKAN DALAM KONTEKS SOSIOLINGUISTIK," Pros. Konf. Linguist. Tah. Atma Jaya KOLITA, vol. 23, no. 23, Sept. 2025, doi: 10.25170/kolita.v23i23.7173.