

Interactive Tool for Javanese Script Transliteration and Corpus Construction

1st A'la Syauqi

Department of Electrical Engineering and Informatics,
Informatics Engineering
Faculty of Science and Technology
Universitas Negeri Malang, Universitas Islam Negeri
Maulana Malik Ibrahim Malang
Malang, Indonesia
ala.syauqi.2405349@students.um.ac.id
syauqi@ti.uin-malang.ac.id

2nd Aji Prasetya Wibawa

Department of Electrical Engineering and Informatics
Universitas Negeri Malang
Malang, Indonesia
aji.prasetya.ft@um.ac.id

3rd Anik Nur Handayani

Department of Electrical Engineering and Informatics
Universitas Negeri Malang
Malang, Indonesia
aniknur.ft@um.ac.id

Abstract—The traditional scriptio continua writing style of Javanese script poses significant challenges for automated transliteration and semantic word segmentation. To address this limitation, this study proposes an interactive human-in-the-loop framework designed for efficient corpus construction. The system integrates a deterministic rule-based transliteration engine with a web-based interface for manual curation. A specialized greedy alignment algorithm featuring a look-ahead mechanism is employed to automatically map curated Latin tokens back to the original Javanese characters. Experimental validation was conducted using a ground truth dataset derived from the 1927 Kejawan periodical. The results demonstrate that the framework achieved a 100% Alignment Success Rate and a negligible processing latency of 0.1 milliseconds per sentence. Additionally, the system successfully extracted 1,992 verified word pairs from 300 sentences. These findings confirm the efficacy of the proposed tool in generating high-quality dual-layer parallel corpora, thereby facilitating the development of advanced neural models for low-resource languages.

Keywords—Javanese Script, Parallel Corpus, Transliteration, Word Segmentation

I. INTRODUCTION

Javanese script is one of Indonesia's traditional writing systems, widely used in historical manuscripts [1], [2], [3], [4] and educational materials [5], [6], [7], [8], [9]. As an abugida [10], [11], each character represents a consonant with an inherent vowel. Traditionally, Javanese texts are written in scriptio continua [12], meaning they are written continuously without spaces between words [10]. This characteristic presents a significant challenge when transliterating the script into Latin. A direct transliteration yields a lengthy sequence of Latin characters without clear word boundaries, rendering the output difficult to parse and comprehend for modern readers.

Although automatic transliteration systems from Javanese to Latin have been developed [13], [14], accurate word segmentation remains a challenge. The primary limitation is linguistic ambiguity, where a sequence of characters can be segmented in multiple ways depending on the context. This task remains difficult for fully automated rule-based or statistical models. Consequently, human intervention is often required to resolve these ambiguities. This study proposes a human-in-the-loop crowdsourcing framework that leverages manual boundary annotation to produce readable text.

Furthermore, this collaborative annotation process facilitates the creation of a high-quality parallel dataset, which is currently scarce for low-resource languages like Javanese.

This study presents an interactive system designed to facilitate word segmentation in Javanese transliteration through user participation. This approach not only enhances text readability but also generates valuable linguistic resources for future research. The main contributions of this work are as follows:

1. The development of an interactive tool for Javanese word segmentation, supporting the curation and alignment of Javanese-Latin parallel sentences.
2. The creation of a verified parallel dataset of words and sentences, accompanied by a corpus of Javanese vocabulary.
3. An experimental validation of the proposed alignment framework to ensure the accuracy and consistency of the generated parallel corpus against a ground truth dataset.

II. LITERATURE REVIEW

Word segmentation for Javanese text has been extensively investigated using various approaches, primarily categorized into rule-based and statistical methods. Deterministic and rule-based methods rely heavily on orthographic constraints and predefined syllabification algorithms [12], [15], [16]. While these methods are computationally efficient and offer high interpretability, they frequently struggle with the complex orthography of Javanese script. This limitation is particularly evident when dealing with ambiguous syllable boundaries and stacked characters, or pasangan, that are not covered by standard rules. To overcome these rigidity issues, statistical machine learning approaches such as Support Vector Machines (SVM) and Conditional Random Fields (CRF) have been introduced [17], [18], [19]. Unlike rule-based systems, these approaches enhance flexibility by employing feature engineering strategies, such as character n-grams and morphological markers, to handle unseen words. However, their reliance on handcrafted features limits their generalization capability, and they still experience significant performance degradation when training data is sparse or domain-specific.

In recent years, deep learning and transformer-based architectures have emerged as a superior alternative by

demonstrating a remarkable capability to capture global context and semantic dependencies for segmentation tasks [20], [21], [22]. For instance, transformer-based canonical segmentation can predict word boundaries by leveraging long-range context from surrounding text, which effectively addresses the ambiguities inherent in scriptio continua. Despite their potential, these data-driven models require large-scale and high-quality annotated datasets to generalize effectively. Unfortunately, such resources are severely limited for low-resource languages like Javanese, and this creates a bottleneck for deploying state-of-the-art models.

Although existing studies have explored algorithmic solutions, there is a notable absence of interactive systems designed to facilitate the creation of such datasets. This study addresses this gap by integrating human expertise with automated processes to produce accurate parallel data while simultaneously improving segmentation quality.

III. METHOD

This section details the architecture of the proposed interactive system, designed to bridge the gap between mechanical transliteration and human readability. The system workflow consists of a two-stage pipeline: (1) an automated transliteration module that converts Javanese script into Latin character sequences based on orthographic rules, and (2) a human-in-the-loop curation interface that allows users to segment these sequences into meaningful words. The final output of this pipeline is a high-quality parallel corpus comprising aligned Javanese-Latin sentences and a verified lexicon of Javanese words.

A. Automated Syllable Transliteration

According to the official guidelines for Javanese orthography, the script comprises several groups of characters, each serving a distinct role in the formation of syllables and words [23]. The primary letters form the core of the writing system and can be written in subjoined forms to indicate consonant sequences within or across syllables, as outlined in Table I. In addition, certain final consonants are represented through dedicated markers. Their orthographic functions are summarized in Table II. Formal registers also employ a set of capital forms used in proper names and institutional terms, as listed in Table III.

TABLE I. *AKSARA NGLEGANA* (PRIMARY LETTERS)

Letter (Aksara)	Subjoined Form (Pasangan)	Latin	Letter (Aksara)	Subjoined Form (Pasangan)	Latin
ꦲ	ꦲꦱꦸꦫꦸꦁ	ha	ꦩ	ꦩꦱꦸꦫꦸꦁ	pa
ꦤ	ꦤꦱꦸꦫꦸꦁ	na	ꦢ	ꦢꦱꦸꦫꦸꦁ	dha
ꦕ	ꦕꦱꦸꦫꦸꦁ	ca	ꦗ	ꦗꦱꦸꦫꦸꦁ	ja
ꦫ	ꦫꦱꦸꦫꦸꦁ	ra	ꦶ	ꦶꦱꦸꦫꦸꦁ	ya
ꦏ	ꦏꦱꦸꦫꦸꦁ	ka	ꦚ	ꦚꦱꦸꦫꦸꦁ	nya
ꦢ	ꦢꦱꦸꦫꦸꦁ	da	ꦩ	ꦩꦱꦸꦫꦸꦁ	ma
ꦠ	ꦠꦱꦸꦫꦸꦁ	ta	ꦲ	ꦲꦱꦸꦫꦸꦁ	ga
ꦱ	ꦱꦱꦸꦫꦸꦁ	sa	ꦧ	ꦧꦱꦸꦫꦸꦁ	ba
ꦮ	ꦮꦱꦸꦫꦸꦁ	wa	ꦠ	ꦠꦱꦸꦫꦸꦁ	tha
ꦲ	ꦲꦱꦸꦫꦸꦁ	la	ꦤ	ꦤꦱꦸꦫꦸꦁ	nga

TABLE II. *PANYIGEG WANDA* (CONSONANTAL DIACRITICS)

Name	Javanese Character	Function in the Orthography	Latin
<i>Wignyan</i>	ꦲꦱꦸꦫꦸꦁ	Indicates the sound <i>h</i> at the end of a syllable.	<i>h</i>
<i>Layar</i>	ꦲꦱꦸꦫꦸꦁ	Indicates the sound <i>r</i> at the end of a syllable.	<i>r</i>
<i>Cecak</i>	ꦲꦱꦸꦫꦸꦁ	Indicates the sound <i>ng</i> at the end of a syllable.	<i>ng</i>
<i>Pangkon</i>	ꦲꦱꦸꦫꦸꦁ	Removes the inherent vowel from the preceding consonant so that it is realized without a vowel.	-

TABLE III. *AKSARA MURDA* (CAPITAL FORMS)

Name	Javanese Character	Function in the Orthography	Latin
<i>Na Murda</i>	ꦲꦱꦸꦫꦸꦁ	Represents the uppercase or emphasized form of the consonant <i>na</i> .	<i>Na</i>
<i>Ka Murda</i>	ꦲꦱꦸꦫꦸꦁ	Represents the uppercase or emphasized form of the consonant <i>ka</i> .	<i>Ka</i>
<i>Ta Murda</i>	ꦲꦱꦸꦫꦸꦁ	Represents the uppercase or emphasized form of the consonant <i>ta</i> .	<i>Ta</i>
<i>Sa Murda</i>	ꦲꦱꦸꦫꦸꦁ	Represents the uppercase or emphasized form of the consonant <i>sa</i> .	<i>Sa</i>
<i>Pa Murda</i>	ꦲꦱꦸꦫꦸꦁ	Represents the uppercase or emphasized form of the consonant <i>pa</i> .	<i>Pa</i>
<i>Ga Murda</i>	ꦲꦱꦸꦫꦸꦁ	Represents the uppercase or emphasized form of the consonant <i>ga</i> .	<i>Ga</i>
<i>Ba Murda</i>	ꦲꦱꦸꦫꦸꦁ	Represents the uppercase or emphasized form of the consonant <i>ba</i> .	<i>Ba</i>

Syllables that begin with a vowel are represented by *aksara swara* (independent vowel characters), which also help clarify the pronunciation of loanwords and proper names. Their functions are detailed in Table IV. Variations of the inherent vowel carried by consonant letters are shown through dedicated *sandhangan swara* (vowel diacritics), which are presented in Table V (Sandhangan Swara).

TABLE IV. *AKSARA SWARA* (INDEPENDENT VOWEL CHARACTERS)

Javanese Character	Description	Latin
ꦲ	Independent vowel representing /a/.	<i>a</i>
ꦤ	Independent vowel representing /i/.	<i>i</i>
ꦩ	Independent vowel representing /u/.	<i>u</i>
ꦕ	Independent vowel representing /e/.	<i>e</i>
ꦫ	Independent vowel representing /o/.	<i>o</i>
ꦱ	Independent vowel representing the schwa /ə/.	<i>ê</i>
ꦠ	Independent vowel representing rhotic schwa /ɾə/.	<i>rê</i>
ꦢ	Independent vowel representing /ɛ/.	<i>lê</i>
ꦠ	Independent vowel representing nasalized /i/.	<i>î</i>
ꦠ	Independent vowel representing nasalized /u/.	<i>û</i>
ꦠ	Independent vowel representing /lɪ/ or /lɔu/.	<i>leu</i>

TABLE V. SANDHANGAN SWARA (VOWEL DIACRITICS)

Name	Javanese Character	Phonetic Value	Transliteration
Wulu	ꦮꦸ	/i/	<i>i</i>
Suku	ꦱꦸ	/u/	<i>u</i>
Taling	ꦠꦭꦶꦁ	/e/	<i>e</i>
Taling Tarung	ꦠꦭꦶꦁꦠꦂꦸꦁ	/o/	<i>o</i>
Pepet	ꦥꦺꦥꦺꦠ	/ə/	<i>ê</i>

Distinct from the final consonant markers, the script also utilizes a specific set of diacritics known as *sandhangan wyanjana* to represent medial consonants or consonant clusters. These markers function similarly to ligature cues in other Brahmic scripts, indicating that a liquid or semivowel is embedded immediately after the initial consonant. The usage of these cluster-forming markers is detailed in Table 6.

TABLE VI. SANDHANGAN WYANJANA (MEDIAL CONSONANT MARKERS)

Javanese Character	Description	Latin
ꦫꦺ	Marks an “r” element attached to the preceding consonant	-r-
ꦶꦫ	Marks a “y” element attached to the preceding consonant	-y-
ꦫꦺꦴ	Represents the vowel form “rê” in Javanese orthography	-rê
ꦠꦭꦶꦁꦠꦂꦸꦁ	Pangkon followed by “wa”, forming a consonant cluster	-w-
ꦠꦭꦶꦁꦠꦂꦸꦁ	Pangkon followed by “la”, forming a consonant cluster	-l-

Based on the orthographic definitions provided in the tables, the transliteration process is implemented using a systematic character-wise mapping algorithm. In this mechanism, the system processes each Javanese character sequentially to determine its corresponding Latin representation based on the surrounding orthographic context. The algorithm initially recognizes a consonantal character and maps it to the Latin equivalent without immediately assigning the inherent vowel. To determine the correct vocalization, the system inspects the subsequent character for any modifying attributes. In cases where no diacritic is present, and the following character is not a subjoined form or pasangan, the system automatically appends the inherent vowel ‘a’ to the consonant. Conversely, if the next character is a pasangan or a vowel-canceling marker, the consonant retains its bare form to reflect the closed syllable structure.

Furthermore, if a specific diacritic or sandhangan is detected, the algorithm applies the corresponding vowel modification directly. This iterative procedure applies to all character types, including Murda letters, to ensure that capital forms and consonant clusters are transliterated accurately. Through this comprehensive examination of diacritics and orthographic markers, the algorithm produces a precise Latin transliteration that preserves the phonetic integrity of the original text. The systematic logic of this character mapping is formalized in Algorithm 1.

ALGORITHM 1: (Rule-Based Javanese Transliteration)**Input:**

Source string of Javanese characters

Output:

Transliterated Latin string

1: **function** TRANSLITERATE(*S*)2: *L* ← empty string

```

3: i ← 0
4: while i < length(S) do
5:   c ← S[i]
6:   cnext ← S[i + 1]
7:
8:   ▷
9:   CASE 1: Consonants (Nglegana) & Capital Forms (Murda)
10:  if IsConsonant(c) or IsMurda(c) then
11:    base ← MapLatinBase(c) ▷ e.g., 'ha', 'Na'
12:    medial ← empty
13:    vowel ← a
14:
15:    ▷ Check for Markers (Cakra, Keret, Pengkal)
16:    if IsOrthographicMarker(cnext) then
17:      medial ← GetMarkerSound(cnext) ▷ e.g., 'r', 'y'
18:      i ← i + 1; cnext ← S[i + 1] ▷ Advance & Look ahead again
19:    end if
20:
21:    ▷ Check for Vowels or Pangkon
22:    if IsSandhanganSwara(cnext) then
23:      vowel ← GetVowelSound(cnext) ▷ e.g., 'i', 'u', 'ê'
24:      i ← i + 1
25:    else if IsPasangan(cnext) or IsPangkon(cnext) then
26:      vowel ← empty ▷ Suppress vowel (closed syllable)
27:    end if
28:
29:    L ← L + base + medial + vowel
30:
31:    ▷ CASE 2: Independent Vowels (Aksara Swara)
32:    else if IsAksaraSwara(c) then
33:      L ← L + MapSwara(c) ▷ e.g., 'A', 'I'
34:
35:    ▷ CASE 3: Final Consonants (Panyigeg Wanda)
36:    else if IsPanyigeg(c) then
37:      L ← L + MapFinal(c) ▷ e.g., 'ng', 'r', 'h'
38:
39:    ▷ CASE 4: Pasangan occurring mid – stream
40:    else if IsPasangan(c) then
41:      L ← L + MapPasangan(c)
42:    end if
43:
44:    i ← i + 1
45:  end while
46:  return L
47: end function

```

As formalized in Algorithm 1, the logic explicitly distinguishes between core consonants (including *aksara murda*), medial consonant markers (*sandhangan wyanjana*), and final consonant markers (*panyigeg wanda*). By distinguishing these categories, the algorithm ensures that complex syllabic structures found in formal Javanese texts are accurately preserved in the Latin output.

B. Interactive Word Segmentation Interface

To facilitate the human-in-the-loop curation process, a web-based interactive interface serves as the intermediate layer between the automated transliteration and the final dataset generation. While the core transliteration engine described in Algorithm 1 generates a strictly continuous phonetic string to preserve orthographic fidelity, the user interface augments this output by applying a preliminary syllabification heuristic. Specifically, the system visualizes the text as a sequence of hyphenated syllables, such as rendering the sequence for *bapak* as “la-jeng-bo-tên”, rather than a continuous stream of characters. This visualization layout, which is depicted in Fig. 1, is designed to aid the annotator in rapidly parsing phonetic structures before making semantic decisions.

Japanese Human-in-the-Loop Tool (Drive Connected)

1. Input Javanese Script

ꦭꦗꦁꦧꦺꦠꦺꦤ꧀ꦏꦺꦁꦒꦺꦠꦤ꧀ꦤꦺꦱꦥꦧꦁꦫꦶꦱꦤꦶ

Transliterate & Visualize

2. Human Curation

Edit hyphens into spaces, 'h' is optional.

lajəng botên kənggêtan nêsêp babar pisan

Align (Add to View)

System Status

Success! Aligned. Data ready to save.

Current Session Data

Sentence Corpus

Word Corpus

Javanese Word	Latin Word
ꦭꦗꦁꦧꦺꦠꦺꦤ꧀	lajəng
ꦧꦺꦠꦺꦤ꧀	botên
ꦏꦺꦁꦒꦺꦠꦤ꧀	kənggêtan
ꦤꦺꦱꦥꦶ	nêsêp
ꦧꦁꦫꦶꦱꦤꦶ	babar
ꦥꦶꦱꦤꦶ	pisan

SAVE (Append to Google Drive CSV)

Use via API

Built with Gradio

Settings

Fig. 1. Proposed Human-in-the-Loop Segmentation Tool.

The operational workflow proceeds through three distinct stages. First, upon receiving the Javanese script input, the system executes the transliteration algorithm and passes the result to a segmentation module. This component automatically inserts hyphens between syllables based on orthographic markers, such as pangkon or final consonants, to create distinct, readable units. Second, the user interacts with the generated Latin text in a dedicated editor panel. The primary task involves establishing meaningful word boundaries by modifying the hyphenated string. Users replace hyphens with spaces to separate words, for instance, changing “*la-jeng-bo-tên*” to “*lajəng botên*”, or remove hyphens to merge syllables within a single word. Finally, once the user confirms the curation, the interface captures the verified text and pairs it with the original script. This action initiates the background alignment process described in the subsequent section, which maps the manual edits back to the specific Javanese characters for corpus construction.

C. Dataset Construction and Word Alignment

The proposed framework generates a dual-layer parallel corpus designed to address data scarcity in low-resource language research. The data is organized into two distinct repositories, each targeting a specific linguistic granularity. The first dataset, archived as `sentence_corpus.csv`, comprises aligned sentence pairs where the source Javanese script is mapped to its Latin transliteration, verified by a human annotator. This resource is critical for training sequence-to-sequence models, such as machine translation systems.

Complementing the sentence-level data, the system automatically derives a granular word-level lexicon, stored in `word_corpus.csv`. Since the annotators provide input at the sentence level, a dedicated alignment mechanism is required to segment the continuous script into discrete lexical units. This process employs a greedy matching strategy, formally defined in Algorithm 2, which systematically maps the continuous Javanese character stream to the corresponding Latin tokens identified during the curation phase.

ALGORITHM 2: (Greedy Matching Word Alignment)

Input:
Javanese string S_{jav} , Human-Annotator Latin string S_{lat}
Output:
List of aligned P

```

1: function ALIGN_WORDS( $S_{jav}, S_{lat}$ )
2:    $P \leftarrow$  empty list
3:    $TargetWords \leftarrow Split(S_{lat}, " ") \triangleright Split\ human\ anotator\ text\ by\ space$ 
4:    $idx \leftarrow 0 \triangleright Pointer\ for\ Javanese\ string$ 
5:
6:   for each word in  $TargetWords$  do
7:      $buffer_{jav} \leftarrow ""$ 
8:      $buffer_{lat} \leftarrow ""$ 
9:      $match\_found \leftarrow False$ 
10:
11:      $\triangleright$ 
12:     while not  $match\_found$  do
13:        $\triangleright Get\ next\ segment\ using\ Logic\ from\ Algorithm\ 1$ 
14:        $seg_{lat}, new\_idx \leftarrow TRANSLITERATE\_SEGMENT(S_{jav}, idx)$ 
15:        $seg_{jav} \leftarrow Substring(S_{jav}, idx, new\_idx)$ 
16:
17:        $buffer_{jav} \leftarrow buffer_{jav} + seg_{jav}$ 
18:        $buffer_{lat} \leftarrow buffer_{lat} + seg_{lat}$ 
19:        $idx \leftarrow new\_idx$ 
20:
21:        $\triangleright Check\ match\ with\ tolerance(Silent - H \& Normalization)$ 
22:       if  $IsMatch(buffer_{lat}, word)$  then
23:
24:          $\triangleright$  CRITICAL: Peek – Ahead Logic
25:          $\triangleright If\ next\ char\ is\ a\ closer(Pangkon/$ 
26:          $PanyigegWanda),\ continue\ loop$ 
27:         if  $IsSyllableCloser(S_{jav}[idx])$  then
28:           continue
29:         end if
30:
31:          $Append(buffer_{jav}, word)$  to  $P$ 
32:          $match\_found \leftarrow True$ 
33:
34:        $\triangleright Safety\ check: If\ buffer\ exceeds\ target,\ alignment\ fails$ 
35:       else if  $Length(buffer_{lat}) > Length(word)$  then
36:         Return Error “Alignment Mismatch”
37:       end if
38:     end while
39:   end for
40:   return  $P$ 
41: end function

```

As detailed in the algorithm, the system iterates through the target Latin tokens and re-processes the Javanese script to identify matching segments. This automated alignment ensures that complex orthographic features, including Pasangan and Sandhangan, are correctly associated with their Latin lemmas without requiring manual tagging at the word level. Consequently, the resulting corpora serve as a rigorous benchmark for evaluating automatic word segmentation models.

The integration of the rule-based transliteration engine, the interactive segmentation interface, and the greedy alignment algorithm constitutes the complete framework proposed in this study. With the methodological foundation and data construction pipeline established, it is essential to validate the system’s efficacy in practical scenarios. The subsequent section presents a comprehensive evaluation of this framework, focusing on the accuracy of the automated alignment logic, the efficiency gains provided by the human-

in-the-loop approach, and the quality of the generated parallel corpora.

IV. RESULT AND DISCUSSION

The efficacy of the proposed framework was validated through a rigorous automated experiment designed to assess the robustness of both the transliteration engine and the word alignment logic. By utilizing a curated ground-truth dataset, the evaluation measures the system's capacity to generate accurate parallel corpora and reconstruct word boundaries consistent with human annotations. The reporting of this evaluation is structured as follows: Subsection A outlines the experimental design, dataset specifications, and the criteria used for success; Subsection B presents the quantitative performance data, specifically focusing on the Alignment Success Rate (ASR); and Subsection C provides a comprehensive analysis of the system's robustness, computational efficiency, and its implication for low-resource language resource creation.

A. Experimental Methodology and Setup

The validation process utilized a ground truth dataset comprising 300 pairs of Javanese script sentences and their corresponding Latin transliterations. These sentences were curated from the historical periodical *Kejawen*, specifically Issue 6, published on February 10, 1927. This source provides linguistically rich content that adheres to standard classical Javanese orthography. The experiment was executed within a Google Colaboratory environment to ensure consistent computational performance and reproducibility.

The experimental procedure functioned as an automated consistency test. The system processed the raw Javanese script from each pair through the proposed alignment framework. It then attempted to reconstruct the word boundaries to match the Latin ground truth. This process evaluates both the coverage of the transliteration rules and the flexibility of the alignment logic without human intervention.

Performance evaluation relied on a strict binary metric based on the algorithm's return value. An alignment is classified as a Pass only if it meets three specific conditions. First, the algorithm must consume the entire Javanese character string without any remainder. Second, every Latin word in the ground truth must find a valid match in the script. This matching process incorporates orthographic tolerance for the normalization of "e" and "ê", as well as the optional presence of the initial consonant "h" regardless of the following vowel sound. Conversely, the system records a failure if it encounters mismatch errors, premature termination where Latin words remain unmatched, or leftover Javanese characters at the end of the process.

To quantify the reliability of the proposed framework, the study employs the Alignment Success Rate (ASR) as the primary evaluation metric. This metric was selected to rigorously measure the system's robustness in bridging sentence-level human annotation with word-level algorithmic segmentation. A high ASR indicates the algorithm's capability to reconstruct valid lexical units without data loss consistently. The ASR is calculated using (1):

$$ASR = \frac{N_{\text{pass}}}{N_{\text{total}}} \times 100\% \quad (1)$$

where N_{pass} denotes the count of sentences correctly aligned according to the criteria above, and N_{total} represents the total number of sentences in the dataset (300). Additionally, the total number of successfully extracted word pairs is recorded to demonstrate the system's efficiency in corpus construction.

B. Performance Evaluation

The automated consistency test yielded quantitative metrics regarding the system's accuracy and computational speed. As summarized in Table VII, the proposed framework achieved a 100% Alignment Success Rate (ASR), successfully reconstructing all 300 sentences from the ground truth dataset without any algorithmic failures.

TABLE VII. EXPERIMENTAL RESULTS

Metric	Value
Dataset Size	300 Sentences
Total Words Harvested	1,992 Words
Successful Alignments	300
Failed Alignments	0
Alignment Success Rate (ASR)	100.00%
Total Execution Time	0.04 seconds
Avg. Processing Time	0.1 ms/sentence

Regarding computational efficiency, the system completed the batch processing in a total of 0.04 seconds, which translates to an average latency of approximately 0.1 milliseconds per sentence. Beyond sentence-level validation, the experiment successfully extracted 1,992 verified word pairs, demonstrating the algorithm's capacity to derive granular lexical data from continuous text inputs. These quantitative findings establish a baseline for assessing the system's robustness in large-scale corpus construction tasks.

C. Discussion

The experimental outcomes collectively validate the proposed framework as a reliable solution for bridging the gap between rule-based logic and human cognition. The attainment of a 100% Alignment Success Rate is primarily attributed to the greedy matching logic augmented with the look-ahead mechanism. By validating the subsequent character in the Javanese script before finalizing a segment match, the algorithm effectively prevents premature token consumption. Moreover, the integration of specific orthographic tolerances proved essential. Specifically, the system accommodates the optional initial consonant "h" regardless of the following vowel sound, as well as the normalization of ambiguous pairs like "e" and "ê". This architectural flexibility ensures that the system accommodates natural variations in Latin transliteration without compromising the precision of the character-to-word mapping. These results confirm that a strictly defined rule-based approach provides sufficient coverage for processing standard classical Javanese orthography.

Regarding computational performance, the negligible processing latency of 0.1 milliseconds per sentence highlights the system's suitability for real-time web applications. In a human-in-the-loop context, system responsiveness is crucial for maintaining user engagement. The low computational cost

ensures that the alignment process runs instantaneously in the background whenever the user updates the text. This efficiency supports the scalability of the tool, suggesting that it can handle larger datasets or concurrent users without requiring significant computational resources.

From a resource generation perspective, the successful extraction of 1,992 verified word pairs demonstrates the framework's capacity to alleviate data scarcity in Javanese Natural Language Processing (NLP). By automating word-level alignment based on sentence-level human curation, the system eliminates the need for manual tag-by-tag annotation. This approach significantly accelerates the construction of granular lexical resources. Consequently, the resulting parallel corpus serves as a foundational asset for developing text-centric downstream applications, particularly neural transliteration models and machine translation systems that require precise orthographic conversion for low-resource languages.

V. CONCLUSION

This study addresses the fundamental challenge of scriptio continua in Javanese script by introducing a human-in-the-loop framework. This system integrates a rule-based transliteration engine with interactive human curation. By leveraging a deterministic "peek-ahead" greedy alignment algorithm, the system effectively bridges the gap between raw phonetic transliteration and semantic word segmentation. Experimental validation using a historical ground truth dataset demonstrated the robustness of this approach. The system achieved a 100% Alignment Success Rate (ASR) while maintaining a negligible processing latency of 0.1 milliseconds per sentence. Furthermore, the automated extraction of 1,992 verified word pairs confirms the system's capacity to streamline the creation of granular lexical resources without requiring manual, tag-by-tag annotation.

In the broader context of low-resource natural language processing, this tool serves as a critical enabler for data digitization and resource construction. The resulting dual-layer parallel corpus comprises both sentence-level and word-level alignments. This comprehensive resource provides a rigorous ground truth essential for training next-generation neural models. Future work will prioritize scaling the dataset through wider crowdsourcing initiatives. Additionally, the generated corpus will be utilized to develop end-to-end deep learning models. These subsequent studies will specifically target automatic Javanese word segmentation, neural transliteration, and machine translation tasks.

REFERENCES

- [1] Alfian Khairur Raaziqin, Lukman Hidayat, and Raden Hadapiningrani Kusumohendarto, "Digital Media In The Preservation of Javanese Script Typeface Cultural Identity," AKSA, vol. 6, no. 1, pp. 859–868, Nov. 2022, doi: 10.37505/aksa.v6i1.73.
- [2] A. Hidayat, "Revitalization of ancient Indonesian characters and the maintenance efforts in past 10 years," LADU, vol. 1, no. 4, pp. 179–186, May 2021, doi: 10.56724/ladu.v1i4.69.
- [3] A. Pandanwangi, S. Himatul Alya, I. Budiman, A. Mochtar Apin, and T. Eka Darmayanti, "Art illuminations in 18th – 19th centuries manuscripts from Ngayogyakarta Hadiningrat Palace as a creative industry development," Cogent Arts & Humanities, vol. 10, no. 2, p. 2277070, Dec. 2023, doi: 10.1080/23311983.2023.2277070.
- [4] A. B. Perdana, "Ragam Langgam Aksara Jawa dari Manuskrip hingga Buku Cetak," MSS, vol. 10, no. 1, p. 1, Aug. 2020, doi: 10.33656/manuskripta.v10i1.140.
- [5] Aderisha Retno Dwi Pangesti and Sukardi, "PASAJA Media for Third Grade Elementary School Students' Javanese Script Writing Skills," j. edutech., vol. 11, no. 2, pp. 203–209, Jan. 2024, doi: 10.23887/jeu.v11i2.65729.
- [6] Endah Sudarmilah, "Theoretical Educational Practices of Siraja Javanese Script Interactive Learning Media Based On 2D Application," kuey, vol. 27, no. 2, pp. 1098–1110, Dec. 2021, doi: 10.17762/kuey.v27i2.255.
- [7] N. H. Insani, H. Hardyanto, and J. Sukoyo, "Facilitating Reading Javanese Letters Skill with a Multimodal Javanese Digital E-Book:," presented at the 5th International Conference on Current Issues in Education (ICCIE 2021), Yogyakarta, Indonesia, 2022. doi: 10.2991/assehr.k.220129.044.
- [8] D. Iskandar, S. Hidayat, U. Jamaludin, and S. Mukti Leksono, "Javanese script digitalization and its utilization as learning media: an etnopedagogical approach," JIJMSed, vol. 1, no. 1, pp. 21–30, June 2023, doi: 10.59965/ijmsed.v1i1.24.
- [9] J. Sukoyo, E. S. Utami, and E. Kurniati, "The Development of Montessori-Based Javanese Script Learning Model:," presented at the 2nd International Conference on Innovation in Education and Pedagogy (ICIEP 2020), Tarakan, Indonesia, 2021. doi: 10.2991/assehr.k.211219.019.
- [10] A. Hidayat, "Revitalization of ancient Indonesian characters and the maintenance efforts in past 10 years," LADU, vol. 1, no. 4, pp. 179–186, May 2021, doi: 10.56724/ladu.v1i4.69.
- [11] D. Krauß, "'Babaring lelakon': The use of '-ing' in Javanese genitive constructions," Wacana, vol. 22, no. 1, p. 182, May 2021, doi: 10.17510/wacana.v22i1.1032.
- [12] A. R. Widiarti, "A method for solving scriptio continua in Javanese manuscript transliteration," Heliyon, vol. 6, no. 4, 2020, doi: 10.1016/j.heliyon.2020.e03827.
- [13] A. R. Widiarti, A. Harjoko, Marsono, and S. Hartati, "The Model and Implementation of Javanese Script Transliteration," in 2017 International Conference on Soft Computing, Intelligent System and Information Technology (ICSIT), Denpasar, Bali, Indonesia: IEEE, Sept. 2017, pp. 51–57. doi: 10.1109/ICSIT.2017.17.
- [14] A. R. Widiarti, R. Pulungan, A. Harjoko, Marsono, and S. Hartati, "A proposed model for Javanese manuscript images transliteration," J. Phys.: Conf. Ser., vol. 1098, p. 012014, Sept. 2018, doi: 10.1088/1742-6596/1098/1/012014.
- [15] L. D. Krisnawati and A. W. Mahastama, "A Javanese Syllabifier Based on its Orthographic System," in Proc. Int. Conf. Asian Lang. Process., IALP, Institute of Electrical and Electronics Engineers Inc., 2018, pp. 244–249. doi: 10.1109/IALP.2018.8629173.
- [16] A. R. Widiarti, "Widiarti-Winarko Algorithm for Grouping Syllables Result from the Javanese Literature Document Image Recognition".
- [17] D. Tanaya and M. Adriani, "Dictionary-based Word Segmentation for Javanese," Procedia Computer Science, vol. 81, pp. 208–213, 2016, doi: 10.1016/j.procs.2016.04.051.
- [18] D. Tanaya and M. Adriani, "Word Segmentation for Javanese Character Using Dictionary, SVM, and CRF," in 2018 International Conference on Asian Language Processing (IALP), Bandung, Indonesia: IEEE, Nov. 2018, pp. 240–243. doi: 10.1109/IALP.2018.8629215.
- [19] B. A. Wicaksono, B. S. Hantono, and T. B. Adji, "Word Segmentation Task for Southeast Asian Abugida Scripts: A Systematic Literature Review," in 2024 2nd International Conference on Technology Innovation and Its Applications (ICTIIA), Medan, Indonesia: IEEE, Sept. 2024, pp. 1–6. doi: 10.1109/ICTIIA61827.2024.10761854.
- [20] M. Nevin, I. K. Agus Ariesta Putra, D. B. Ansori, R. Sarno, and A. Haryono, "A Deep Learning Approach for Word Segmentation in Javanese Letter Manuscript Transliteration," in 2024 2nd International Conference on Technology Innovation and Its Applications (ICTIIA), Medan, Indonesia: IEEE, Sept. 2024, pp. 1–6. doi: 10.1109/ICTIIA61827.2024.10761167.
- [21] S. H. Wijono, M. R. Alhamidi, M. H. Hilman, and W. Jatmiko, "Canonical Segmentation Using Affix Characters as a Unit on Transformer for Javanese Language," in 2021 6th International Workshop on Big Data and Information Security (IWBIS), Depok, Indonesia: IEEE, Oct. 2021, pp. 67–72. doi: 10.1109/IWBIS53353.2021.9631839.
- [22] S. H. Wijono, K. Azizah, and W. Jatmiko, "CANONICAL SEGMENTATION FOR JAVANESE-INDONESIAN NEURAL MACHINE TRANSLATION," Journal of Engineering Science and Technology, vol. Special Issue on AASEC2022, August (2022), pp. 62–68, Accessed: November 26, 2025. [Online]. Available:

https://jestec.taylors.edu.my/Special%20Issue%20AASEC2022/RPI_SI_2023_9.pdf.

- [23] Compilation Team, Guidelines for Writing Javanese Script. Yogyakarta: Yayasan Pustaka Nusantara, 1995.