A method for solving scriptio continua in Javanese manuscript transliteration

Anastasia Rita Widiarti a,*, Reza Pulungan b

a Department of Informatics, Faculty of Science and Technology, Universitas Sanata Dharma, Yogyakarta, Indonesia
b Department of Computer Science and Electronics, Faculty of Mathematics and Natural Sciences, Universitas Gadjah Mada, Yogyakarta, Indonesia

ARTICLE INFO

Keywords:
Computer Science
Scriptio continua
Javanese manuscript
Syllables concatenation
Transliteration
Greedy

ABSTRACT

Many Javanese manuscripts in Indonesia are stored in museums and libraries. Most of these manuscripts were written using local scripts that are rarely used in everyday life, and hence a software application that can help and improve the reading of these manuscripts is valuable. An essential step in automatic manuscript image transliteration is post-processing, which involves editing and concatenating syllables into words. The main problem of post-processing is that there exists no symbol for space between words in a sentence, which is called the scriptio-continua problem. This paper proposes methods based on the backtracking algorithm to solve the scriptio continua in the post-processing step of Javanese manuscript image transliteration. The proposed methods use a depth-first search in selecting relevant candidate words to determine whether to merge a new syllable or not. The results of the proposed methods to concatenate 17,687 syllables from the Hamong Tani book using a dictionary containing 49,801 words are found to be satisfactory in terms of computation and accuracy. The accuracy of the implemented greedy and brute-force methods is both 81.64%. However, the greedy-based method is more efficient and has a better performance than the brute-force method.

1. Introduction

Indonesia has a long and rich cultural heritage captured in the form of information about the noble culture of the past in either scripts or manuscripts that remained uncounted until recently [1]. These manuscripts contain historical information about worship, manners, history, folklore, traditional technologies, spells, genealogy, talismans, poetry, politics, governments, laws, customs, traditional medicine, and culture [2]. The medium of the manuscripts takes various forms. They were written in many different languages using different scripts [3, 4, 5, 6]. Many ancient manuscripts are still stored well in various museums or libraries in Indonesia. Automatic reading aids are valuable to help simplify and speed up the reading of these manuscripts. In this way, more generations of Indonesians will be able to use the important knowledge contained in the manuscripts.

An automatic transliteration of the manuscripts is an opportunity to ease the reading of those manuscripts. Manuscript transliteration is generally conducted by reading the manuscript and then writing the results to another piece of paper or storing them in a computer using a specific text-processing program. The procedure of transliteration presupposes that the workers fully understand how to read the manuscript, that no consideration is given to the length of the work, and that workers will concentrate sufficiently to minimize errors in the rewriting [7].

Many researchers have been working on automatic manuscript transliteration in various scripts using various methods. Shridhar and Kimura [8], in their seminal work, proposed an automatic transliteration method of handwritten words in Roman script, which comprises three main steps, i.e., pre-processing document, segmenting scripts in the document, and recognizing each of the scripts forming the words. These similar steps have been applied for Javanese manuscript transliteration as part of the Javanese optical character recognition (OCR) project [9, 10, 11, 12].

Javanese scripts are part of the Brahmi script group, which has been widely used in Southeast Asian societies in the past. Javanese scripts are different from alphabetical scripts because one Javanese letter represents a consonant or a combination of a consonant and a vowel, which results in two or more letters when transliterated into the Roman script. Since this transliteration results are in the form of syllables, this way of writing Javanese letters is called alphasyllahary or abugida [13, 14]. As an abugida writing system, a unit of Javanese script represents a consonant-vowel sequence based on the consonant letter. Each consonant has an inherent vowel that can be changed by diacritics or other
modifications. One characteristic shared by this writing system is that there is no word delimiter.

In most of the classic Javanese manuscripts, many symbols stand as punctuation marks, including “aded-aded” for the beginning of a sentence, “pada lungsi” as a full stop, “pada lingsa” as a comma, and “pada” as a symbol marking the beginning, middle, or end of the letters or lyrics of songs. The complete process of segmenting and recognizing Javanese scripts and these punctuation symbols has been previously discussed by Widiarti et al. [12]. Among those symbols, however, there is no space symbol that separates words. The results of the transliteration of a Javanese manuscript image, therefore, will be a text document containing a sequence of syllables without spaces.

The spaceless writing model has been going on since the 20th century AD [15]. Various regions in Indonesia and other parts of the world, such as Bengkulu, Lampung, Bali, Japan, China, Cambodia, Laos, Tibet, Vietnam, and Roman manuscripts, also used the rule for writing manuscripts without spaces [16]. Such a way of writing without spaces is called scriptio continua. With the scriptio-continua writing model, the process of forming a sentence consisting of a sequence of words in each sentence becomes a problem because there are no punctuation marks that stand as a marker of the distance between words.

Scriptio continua typically happened because the manuscript’s writer (or scribe) had to continue writing the content of the manuscript dictated by the author, who was continuously providing information that the scribe did not want to miss. In earlier times, oral tradition was more prominent, and the writer and author of a manuscript were usually different people. By writing scripts continuously, there is also another advantage to be gained, namely that the readers are given full flexibility to interpret the groups of words according to the context of the conversation they caught [17].

This paper presents the results of our research in exploring methods for forming words out of syllables produced by the transliteration process of Javanese-language manuscripts as a solution to the scriptio-continua problem. Two algorithms are developed and implemented, namely based on brute-force and greedy strategies. In principle, both algorithms use a depth-first search in processing the syllables. We describe the results of the implementation and then compare the two algorithms. Our aim in this paper is to develop a fast and efficient algorithm for segmenting contiguous syllables into words.

The rest of the paper is organized as follows: Section 2 reviews related work in word segmentation. Section 3 describes the core ideas of the proposed algorithm. Section 4 details the data used to evaluate the proposed method. Section 5 presents the result of our experiments and discussion. The last section concludes the paper.

2. Related work

Many researchers in various countries have been working on the post-processing step in the automatic transliteration of manuscripts. Rhodes compared the baseline dictionary model and conditional random field segmentation model for segmenting words from the notes of Roman poets [15]. Palmer and Burger [18] presented the results of their experiments on Chinese word segmentation with a characters-as-word method, maximum matching algorithm, and NMSU segmenter. Aroonmanakun and Rivipoon [19] used a statistical approach to solve the problem of word segmentation. This research was part of a project to transliterate the Thai language writing system to Roman scripts.

Jurgens and Steven discovered that compared to the lexical feature-based approach, word sense induction (WSI) based cluster is very sensitive in finding differences in the sense of the word from the context [20]. Krovetz and Croft performed an analysis of lexical ambiguity in the information retrieval to see the usefulness of the meanings of words on the election process to the relevance of documents found. They found that weighting based on the number of meanings of those words improves the effectiveness of information discovery data collection, although relatively few were used [21].

From earlier studies in Indonesia, researches related to the analysis of words have been conducted. Sucidi carried out a research on the syntactic and semantic analysis of interpretation of natural language processing (NLP) and found that there is no method of parsing which is ideal for all kinds of problems in NLP, that thematic roles further clarify the roles of each element of a sentence, and that word-sense hierarchy used by the selectional restrictions helps carry out the word-sense disambiguation process [22]. Margareth et al. found that latent semantic analysis (LSA) can understand various semantic information with relatively little data on training [23].

Regarding the final process in recognizing Javanese manuscripts, Widiarti and Winarko [24] proposed an algorithm to combine syllables generated from the Javanese script transliteration into words. In another useful study by Widiarti and Pulungan [25], discrete-time Markov chains (DTMC) are used to predict the syllables that would appear following a given syllable in word-groups of the Javanese language. This study is useful when deadlocks in syllable grouping appear as a result of Javanese script transliteration. The prediction information is essential because when the quality of the original document to be recognized is low, difficulties in recognition of the script images in the document may arise. In certain circumstances, it may be necessary to guess the content of a script image by using the knowledge built using the DTMC application described above. Agusta [26] studied the creation of a stemming algorithm using 30 Javanese-language texts from 30 documents. The resulting Agusta-Harjoko (AH) stemming algorithm still has some limitations in terms of partial words, and its success depends on the completeness of the used dictionary.

Recent researches in the field of OCR indicate that the three steps mentioned before—pre-processing document, segmenting scripts in the document, and recognizing all scripts forming the words—are common but given different terms. For instance, they are used for OCR of Urdu scripts [27]. Furthermore, researches on post-processing are also becoming more prevalent in other cases because post-processing is considered to be a possible way to minimize errors or increase improvement. Mei et al. [28] used a statistical approach for the post-processing step, while still including the role of humans to minimize mistakes. Mokhtar et al. [29] introduced an OCR post-processing method based on neural machine translation (NMT). Nguyen et al. [30] explored possible errors in OCR and then presented an idea of error classification to improve the performance of post-OCR processing.

3. Proposed method

Two approaches may be used in developing a syllable concatenating method, namely forward and backward approaches, as applied in a tree data structure. For each approach, a dictionary becomes an essential part as a reference whether each candidate word produced is valid or not as a word.

The forward approach assumes that every time a new syllable is found in the input data, the new syllable is made as root. All of the next possible syllables that will become children are the results of the syllable segmentation of candidates in the dictionary for all words beginning with the root. In this approach, every time a new syllable is to be processed, the tree structure must always be adjusted with the appropriate number of children depending on how many possible words in the dictionary begin with the new syllable. Out of such a root, there will be many possible branches, most of which will not be used because only one word is valid.

This approach can be simplified by creating a tree from the other way around, namely by a backward approach. The tree can be seen from the leaves of the possible candidate tree, which is taken from the last syllable from the result of the manuscript transliteration. The hope is that, in such a tree, there are fewer branches that lead to leaves so that the number of paths to trace is less.

This research proposes a forward approach with a modified brute-force principle. The approach taken is focused on ensuring that the
results of concatenating syllables must form as many legitimate words as possible. Legitimate means that not only is the word in the dictionary, but the word that is most suitable for the corresponding manuscript has been selected.

The main principle of word-formation from syllable results in general in this study is based on an algorithm for word segmentation proposed by Erickson [16]. Every time a character is read from the input string, after merging the characters, the algorithm checks whether the merging results in a new word. If a new word is formed, it then checks whether the next character and so on until the last character are still possible to form a word.

The strategy we propose to ensure that all words have been produced is using the dictionary to list all possible words that can be produced from the syllables found in the manuscript. The unit of data input is a syllable obtained from one Javanese script image transliteration or a grouping of Javanese scripts that are considered as one single syllable. Our first proposed syllables concatenation algorithm, which is based on this brute-force strategy, is depicted in Algorithm 1.

Algorithm 1 The brute-force syllables concatenation algorithm.

| Input: syllables: list of syllables, dict: list of valid words |
| Output: words: list of predicted words |
| 1: function ORIGNAL-SYLL-CON(syllables, dict): words |
| 2: words = ∅ |
| 3: for i = 0 to length(syllables) do |
| 4: cand ← "" |
| 5: for j = i to length(syllables) do |
| 6: if isValid(syllables[j]) then |
| 7: cand ← concatenate(cand, syllables[j]) |
| 8: if (length(cand) ≤ longest and dict.contains(cand)) then |
| 9: words ← add(cand) |
| 10: end if |
| 11: end if |
| 12: end for |
| 13: end for |
| 14: return words |
| 15: end function |

The syllable concatenation is carried out every time a next syllable is processed. Starting from the first syllable found in the manuscript, the algorithm each time tries to concatenate the syllables found so far with the next syllable. The next step is to check whether the result of concatenating those syllables forms a valid word. A word is declared valid if it is in a dictionary. The process of concatenating syllables for each syllable will end if the length of the new word formed is the same as the longest word that begins with the initial syllable in the dictionary.

Algorithm 1 uses a brute-force strategy because it concatenates existing syllables with the next one regardless of whether the result forms a valid word or not. The process of concatenations continues until the length of the resulting string in the output candidate word is maximal, which is defined as the length of the longest word in the dictionary. This strategy will definitely find a word, but many iterations may be required to find it.

The brute-force algorithm above certainly guarantees that all formed words are correct. However, this method has a quadratic complexity in the number of syllables processed, which makes the computation time of its implementation very large. The brute-force algorithm can be improved by running a greedy strategy. During the intermediate word formation, words can be immediately checked in the word dictionary, starting with the first syllables that have clearly been known to the maximum length of words in the dictionary. This strategy presupposes that every time a new word is found, the next syllable combination is focused on the part of the word that contains the previously found word so that the correct word is expected to be obtained. Algorithm 2 shows the modification of Algorithm 1, whose word-forming strategy has been changed.

Algorithm 2 The greedy syllables concatenation algorithm.

| Input: syllables: list of syllables, dict: list of valid words |
| Output: words: list of predicted words |
| 1: function GREEDY-SYLL-CON(syllables, dict): words |
| 2: words = ∅ |
| 3: for i = 0 to length(syllables) do |
| 4: cand ← "" |
| 5: for j = i to length(syllables) do |
| 6: if (isValid(syllables[j]) then |
| 7: cand ← concatenate(cand, syllables[j]) |
| 8: if (length(cand) ≤ longest) then |
| 9: if (checkStarWithDict(cand)) then |
| 10: if (dict.contains(cand)) then |
| 11: words ← add(cand) |
| 12: end if |
| 13: end if |
| 14: break for-j |
| 15: end if |
| 16: end if |
| 17: break for-i |
| 18: end if |
| 19: end if |
| 20: end for |
| 21: end for |
| 22: return words |
| 23: end function |

The input of the proposed syllables concatenation algorithms is a sequence of syllables obtained from Javanese manuscript transliteration. There is no limit on the maximum number of syllables that the method can process. Every time the method is run, a dictionary of words is needed as a reference for the correctness of the words formed.

Dictionary, in this paper, is simply meant a list of valid words. This dictionary is formed by collecting all Javanese words that appear in “Bausastra,” an established Javanese-Indonesian dictionary [31]. These include all defined words and also all words in the definition of each word. For example, the definition of the word “harmonika” is “jinisir piranti musik sebul sing gedhene sagegeman tangan, carane nguneke kanthi nyebul bolongan nadha sing ditempelake ing lambe”. We add “sagegeman” into our dictionary because this word has not appeared anywhere beforehand.

In our experiments, we use all syllables from page 22 until page 77 of the transliteration result of the Hamong Tani book [32]. Not all pages of the Hamong Tani book are used because of the limited quality of the pre-processing results, which is not directly related to the post-processing steps we are dealing with in this paper. The Hamong Tani book was selected as a sample because it is commonly used as a research sample in many studies of Javanese literature. The book is also available in our university’s library and is well maintained.

Page 22 of the Hamong Tani book used in this research is shown in Fig. 1. There are 336 Javanese characters on the page. The result of the transliteration should also be 336 syllables because each script in Javanese manuscripts represents a syllable consisting of one or more letters. However, some characters also symbolize punctuation, so from 336 characters on page 22, only 263 syllables are ready to be formed into words.

Since some words in the Hamong Tani book do not occur in the Bausastra dictionary, all words in the Hamong Tani book are also stored in the dictionary. An example is the word “anyjarwani,” which does not appear in the Bausastra. The total number of words that have been included in the dictionary is 49,801 words, which consist of all words in the Bausastra and all words in the Hamong Tani book.
null pun siti hingkang katanenman sata kama wopunika sakalangkung kahot tipu katimbang kalayan pemadelling siti hingkang kahelettelettan pananenmipun wahu hingkang punika kados pundi katerangnganipun hing ngataassis punika serat nget nga tani hing tembung walandi sampun hanyjarwani kados hingkang bu hing ngandhap punika adeg-adeg sakatadhing tanem tuwhu punika sa sa pagesangnganipun hingkang pinanggih won ten hing siti da bebada nipun pagesangngan wahu haiwit saiking warnining tatanenman kados tes ta hingkang katanem sapijan ded dan punika saasampunning kahunduhu huwohhipun siti hingkang tilas katanenman ded dan wahu hing tembe badhe dha suka pagesangngan dhateng sa sipun tanem tutu mila manawi tanemmangipun wahu ten kasantunnan hinggih badhe dha hawon hing pa

Fig. 1. Page 22 of the Hamong Tani book [32].

5. Results and discussion

The proposed methods are implemented in Java. To manage the word dictionary data, MySQL is used. In its development, the dictionary can always be updated if there are other words that have not been input, or if new words are found. If the implementation is to be further developed for other languages, it only needs to update or input the words of the language.

Before running the syllables concatenation algorithms, all syllables that are not meaningful have been removed or transformed into new syllables that contribute to making words. For example, if a script related to punctuation marks, like symbols for the beginning of a word, is found, then it is skipped. The necessary preparation of the sequence of syllables has been previously discussed in [12]. After obtaining the syllables that are ready to be processed into words, the syllables are then stored in a text file to become input data to the implementation of the proposed algorithms. Fig. 2 shows an example of input data for the syllables concatenation algorithms. It is the transliteration result of the Hamong Tani book page 22 and has been pre-processed beforehand.

The implementation of the two algorithms produces the same output, which is in the form of a sequence of words. For the input data in Fig. 2, Fig. 3 depicts the 96 words formed based on the words in the dictionary. Both algorithms produce exactly the same words.

Evaluation of the output is focused on calculating the number of correct words that should appear as a result of the transliteration. A word is correct if the syllabic composition of the word and its location in the sentence match the arrangement established by a philologist. This arrangement is called the ground truth, and it is fixed prior to the experiments. We also check if there are missing syllables during the running of the algorithms.

The accuracy of the algorithms for a particular page is the ratio between the number of valid words found by the algorithms and the number of words in the ground truth. Once all experiments are completed, we then evaluate the results of the proposed methods and compare them with the ground truth previously established by the philologist to determine the accuracy of the proposed methods for each page. For example, as a result of the transliteration of page 22, 96 words are produced by the system. However, there are only 78 correct words among those words. Therefore, the accuracy of the syllables concatenation algorithms for that page is 81.25%. Tables 1 and 2 show the results of applying the proposed algorithms on pages 22-77 of the Hamong Tani book.

Of the 5,008 words that should be formed, both algorithms obtain 5,744 valid words, namely those that are present in the dictionary. From the 5,744 valid words, after being traced word-by-word in order, 4,089 words are correct, namely, according to the ground truth. Therefore, the accuracy of the algorithms is 81.64%.

The word-formation error is still quite large, almost 19%. Our evaluation, together with the philologist, reveals three causes: First, pre-processed data even before the syllables concatenation process contains
Table 1

The results of experiments on the syllables concatenation algorithms for pages 22-50 of the Hamong Tani book.

| Page | Number of syllables | Number of correct words | Number of words | Accuracy |
|------|---------------------|-------------------------|-----------------|----------|
| 22   | 336                 | 78                      | 96              | 81.25    |
| 23   | 335                 | 83                      | 100             | 83.00    |
| 24   | 346                 | 74                      | 98              | 75.51    |
| 25   | 343                 | 75                      | 97              | 77.32    |
| 26   | 334                 | 91                      | 101             | 90.10    |
| 27   | 353                 | 91                      | 100             | 91.00    |
| 28   | 334                 | 82                      | 102             | 80.39    |
| 29   | 345                 | 91                      | 105             | 86.67    |
| 30   | 317                 | 78                      | 101             | 77.23    |
| 31   | 303                 | 74                      | 94              | 78.72    |
| 32   | 317                 | 73                      | 94              | 77.66    |
| 33   | 308                 | 70                      | 93              | 75.27    |
| 34   | 338                 | 83                      | 99              | 83.84    |
| 35   | 246                 | 53                      | 68              | 77.94    |
| 36   | 243                 | 49                      | 68              | 72.06    |
| 37   | 208                 | 42                      | 59              | 71.19    |
| 38   | 212                 | 43                      | 62              | 69.35    |
| 39   | 227                 | 45                      | 66              | 68.18    |
| 40   | 207                 | 49                      | 63              | 77.78    |
| 41   | 334                 | 82                      | 94              | 87.23    |
| 42   | 267                 | 64                      | 78              | 82.05    |
| 43   | 191                 | 49                      | 56              | 87.50    |
| 44   | 324                 | 68                      | 94              | 72.34    |
| 45   | 258                 | 61                      | 76              | 80.26    |
| 46   | 326                 | 68                      | 93              | 73.12    |
| 47   | 346                 | 78                      | 92              | 84.78    |
| 48   | 350                 | 78                      | 98              | 79.59    |
| 49   | 342                 | 79                      | 99              | 79.80    |

Table 2

The results of experiments on the syllables concatenation algorithms for pages 51-77 of the Hamong Tani book.

| Page | Number of syllables | Number of correct words | Number of words | Accuracy |
|------|---------------------|-------------------------|-----------------|----------|
| 50   | 219                 | 57                      | 65              | 87.69    |
| 51   | 335                 | 91                      | 104             | 87.50    |
| 52   | 338                 | 86                      | 101             | 85.15    |
| 53   | 138                 | 41                      | 47              | 87.23    |
| 54   | 333                 | 90                      | 107             | 84.11    |
| 55   | 256                 | 83                      | 111             | 74.77    |
| 56   | 143                 | 37                      | 44              | 84.09    |
| 57   | 290                 | 66                      | 88              | 75.00    |
| 58   | 184                 | 47                      | 55              | 85.45    |
| 59   | 130                 | 32                      | 41              | 78.05    |
| 60   | 334                 | 72                      | 87              | 82.76    |
| 61   | 309                 | 86                      | 102             | 84.31    |
| 62   | 282                 | 58                      | 86              | 67.44    |
| 63   | 318                 | 72                      | 88              | 81.82    |
| 64   | 314                 | 77                      | 94              | 81.91    |
| 65   | 353                 | 94                      | 106             | 88.68    |
| 66   | 349                 | 88                      | 106             | 83.02    |
| 67   | 398                 | 100                     | 129             | 77.52    |
| 68   | 346                 | 89                      | 109             | 81.65    |
| 69   | 353                 | 96                      | 114             | 84.21    |
| 70   | 340                 | 93                      | 103             | 90.29    |
| 71   | 324                 | 92                      | 101             | 91.09    |
| 72   | 336                 | 96                      | 103             | 93.20    |
| 73   | 352                 | 87                      | 106             | 82.08    |
| 74   | 284                 | 77                      | 89              | 86.52    |
| 75   | 350                 | 81                      | 99              | 81.82    |
| 76   | 243                 | 69                      | 75              | 92.00    |
| 77   | 347                 | 81                      | 102             | 79.41    |

standardization of Javanese writing. These errors can be corrected by checking the rules applied in the syllables pre-processing and improving the collections of words in the dictionary.

The brute-force strategy applied in Algorithm 1, which always lists all words that might be formed from the current concatenated syllables every time a new syllable is processed, is undoubtedly very inefficient. The result of appending the next syllable to the word formed (from the previous syllables) so far is not taken into consideration in the next iteration. Indeed, this guarantees that a valid word must be found. However, with the number of syllables of 17,687, for example, the number of iterations swells from 17,687 to 191,640. This is 10.835 times the number of syllables being processed.

To improve on the principle of incorporation used in the first algorithm, a mechanism of not re-reading syllables that have become part of the word is developed for the concatenation process of syllables. Another change made to the algorithm is the use of function check-StartWith(), which checks whether there are words in the dictionary that start with the current concatenated syllables as an implementation of the greedy strategy. With the development of the second algorithm, to combine 17,687 syllables can be accomplished in only 55,669 iterations or 3.147 times the number of syllables, which is a significant reduction compared to the first algorithm. Table 3 shows the computation times as well as the number of iterations of the two algorithms for concatenating all syllables in pages 22-77 of the Hamong Tani book.

Changes made to improve the performance of the first algorithm result in a decrease in the number of iterations, which directly affected the computation time of the algorithms, as shown in Table 3. There is a significant improvement in the computation time because the iterations are significantly fewer with the greedy algorithm.

6. Conclusion

The syllable concatenation algorithms, which form a post-processing step in the transliteration of accented Javanese manuscript images with a forward approach, can be used to complete the final stage of transliteration of Javanese-style manuscript images. The accuracy of the proposed algorithms is above 81%, which indicates that they are adequate for practical use. The algorithms can also be applied to the transliteration process of other manuscripts written in different scripts, merely by replacing the word dictionary used.

The disadvantage of these algorithms, both the brute-force and the greedy approaches, is that a dictionary remains the critical source of success in concatenating syllables. If a new word found that is not in the dictionary, an error is sure to occur. Future research direction could be to improve the performance of the proposed algorithms by combining them with artificial intelligence and exploring how users can participate in updating the dictionary with new words to continuously improve the accuracy of the methods.

Declarations

Author contribution statement

A.R. Widiarti, R. Pulungan: Conceived and designed the experiments; Performed the experiments; Analyzed and interpreted the data; Contributed reagents, materials, analysis tools or data; Wrote the paper.
Funding statement

This research is funded by the Ministry of Research, Technology and Higher Education of the Republic of Indonesia, Penelitian Pascakdoctor scheme, grant no. 051/HB-LIT-IV/2017 and 010/HB-LIT/II/2018.

Competing Interest statement

The authors declare no conflict of interest.

Additional information

No additional information is available for this paper.

Acknowledgements

The authors wish to thank Universitas Sanata Dharma, Universitas Gadjah Mada, and the programmer team in this project: Robertus Adi Nugroho, Eduardus Hardika Sandy Atmajaya, Kartono Pinaryanto, and all informatics USD students who have joined this dictionary project. The authors also thank Yohannes Widodo, S.S., a philologist and librarian at Universitas Sanata Dharma, for providing the ground truth in our experiments.

References

[1] S. Paku Buwana V. Cenhtini: Tambangerais-Amongraga Jilid IV, Gadjah Mada University Press, Yogyakarta, 2010 (in Bahasa Indonesia).
[2] D.L. Sukawati, Inventarisasi naskah lama Madura (Inventorization of ancient Madurese manuscripts), Manuskripta 1 (2) (2011) 17–30 (in Bahasa Indonesia).
[3] T.E. Behrend, Katalog Induk Naskah-Naskah Nusantara Jilid 1: Museum Sonobudoyo Yogyakarta (Master Catalogue of Nusantara Manuscripts, vol. 1), Sonobudoyo Museum, Yogyakarta, Jakarta, 1990 (in Bahasa Indonesia).
[4] T.E. Behrend, T. Pudjiastuti, Katalog Induk Naskah-Naskah Nusantara Jilid 3-A: Fakultas Sastra Universitas Indonesia (Master Catalogue of Nusantara Manuscripts, vol. 3-A: Faculty of Literature University of Indonesia), Yayaasan Obor Indonesia, Jakarta, 1997 (in Bahasa Indonesia).
[5] E. Budiarti, S.E.P. Adj, K. Setyawan, Y.A. Rahayu, Karac: jejek-jejek perjalanan keilmiahan Zoetmulder (Karac: traces of Zoetmulder’s scientific journey), Penerbit Universitas Sanata Dharma, Yogyakarta, 2008 (in Bahasa Indonesia).
[6] N. Syahrul, Upaya dan penyelamatan naskah kuno Lampung (Efforts in conserving ancient Lampungnese manuscripts), Manuskripta 1 (2) (2011) 1–16 (in Bahasa Indonesia).
[7] A.R. Widiarti, A. Harjoko, Marsono, S. Hartati, The model and implementation of Javanese script image transliteration, in: 2017 International Conference on Soft Computing, Intelligent System and Information Technology (ICSIIT), 2017, pp. 51–57.
[8] M. Shiridhar, F. Kimura, Segmentation-based cursive handwriting recognition, in: Handbook of Character Recognition and Document Image Analysis, World Scientific, 1997, pp. 123–156.
[9] A.R. Widiarti, A. Harjoko, Pengelajuan citra dokumen sastra Jawa, studi kasus pada buku sastra Jawa: Menak Sorangan 1 dan Panji Sekar (Image recognition of Javanese manuscripts, case studies on Javanese literatures: Menak Sorangan I and Panji Sekar), Sains dan Sibernetika 19 (2006) 269–282 (in Bahasa Indonesia).
[10] A.R. Widiarti, Model transliterasi otomatis citra naskah aksara Jawa (A model for automatic transliteration of images of Javanese manuscripts), PhD thesis, Universitas Gadjah Mada, Yogyakarta, 2015 (in Bahasa Indonesia).
[11] A.R. Widiarti, A. Harjoko, Marsono, S. Hartati, Preprocessing model of manuscripts in Javanese characters, J. Signal Inf. Process. 5 (2014) 112–122.
[12] A.R. Widiarti, R. Pulungan, A. Harjoko, Marsono, S. Hartati, A proposed model for Javanese manuscript images transliteration, J. Phys. Conf. Ser. 1098 (2018) 012014.
[13] W. Bright, A matter of typology: alphasyllabaries and abugidas, Writ. Lang. Lit. 2 (1) (1999) 45–55.
[14] C. Court, The spread of Brahmi script into Southeast Asia, in: P.T. Daniels, W. Bright (Eds.), The World’s Writing System, Oxford University Press, New York, 1996, pp. 445–449, Ch. 41.
[15] D. Rhodes, Conditional random field Latin word segmenter, Tech. rep., University of Stanford, 2013.
[16] J. Erickson, Backtracking, in: Algorithms, University of Illinois at Urbana-Champaign, 2019, pp. 71–96.
[17] P. Worlesy, S. Supomo, M. Fletcher, T. Hunter, Kakawin Sumanasaaantaka mati karena Bunga Sumanas karya Mpu Monaguna: kajian sebuh pusi ejaw Kawa kuno (Kakawin Sumanasantaka dies because of Bunga Sumanas by Mpu Mona-guna: study of an old epic Javanese poetry), Yayasan Obor Indonesia, Jakarta, 2014 (in Bahasa Indonesia).
[18] D.D. Palmer, J. Burger, Chinese word segmentation and information retrieval, Tech. rep., Association for the Advancement of Artificial Intelligence (AAAI), 1997, Technical Report SS-97-05.
[19] W. Arowmonanakun, W. Rivepboon, A unified model of Thai Romanization and word segmentation, in: Proceedings of the 18th Pacific Asia Conference on Language, Information and Computation, Logico-Linguistic Society of Japan, Waseda University, Tokyo, Japan, 2004, pp. 205–214.
[20] D. Jurgens, K. Stevens, Measuring the impact of sense similarity on word sense induction, in: Proceedings of the First Workshop on Unsupervised Learning in NLP, Association for Computational Linguistics, Edinburgh, Scotland, 2011, pp. 113–123.
[21] R. Krovetz, W.B. Croft, Lexical ambiguity and information retrieval, ACM Trans. Inf. Syst. 10 (2) (1992) 115–141.
[22] J. Sucidiani, Studi analisis metode-metode parsing dan interpretasi semantik pada natural language processing (Analytical study of parsing methods and semantic interpretation on natural language processing), J. Inform. 2 (1) (2001) 13–22 (in Bahasa Indonesia).
[23] E. Margaretha, Franky, R. Manurung, English-to-Indonesian lexical mapping using latent semantic analysis, in: Proceedings of the 2nd International, MALINDO Workshop, Cyberjaya, Malaysia, 2008, pp. 1–7.
[24] A.R. Widiarti, E. Winarko, Widiarti-Winarko algorithm for grouping syllables result from the Javanese literature document image recognition, in: 16th WSEAS International Conference on Computers, Kos Island, Greece, 2012, pp. 266–271.
[25] A.R. Widiarti, R. Pulungan, Aplikasi DTMC untuk post-processing pengelajuan citra dokumen teks (Application of DTMC for post-processing of textual documents’ image recognition), in: Prosiding Konferensi Nasional Sistem Informasi (KNSI), STIKOM STIKOM, Bali, 2012, pp. 830–835 (in Bahasa Indonesia).
[26] L. Agusta, Algoritma stemming untuk dokumen teks bahasa Jawa (Stemming algorithms for Javanese textual documents), Master’s thesis, Universitas Gadjah Mada, Yogyakarta, 2009 (in Bahasa Indonesia).
[27] N.H. Khan, A. Adnan, Urdu optical character recognition systems: present contributions and future directions, IEEE Access 6 (2018) 46019–46046.
[28] J. Mei, A. Islam, A. Moh’d, Y. Wu, E. Milios, Statistical learning for OCR error correction, Inf. Process. Manag. 54 (6) (2018) 874–887.
[29] K. Mokhtar, S.S. Bukhari, A. Dengel, OCR error correction: state-of-the-art vs an NMT-based approach, in: 2018 13th IAPR International Workshop on Document Analysis Systems (DAS), 2018, pp. 429–434.
[30] T. Nguyen, A. Jaitwor, M. Coutatay, N. Nguyen, A. Doucept, Deep statistical analysis of OCR errors for effective post-OCR processing, in: 2019 ACM/IEEE Joint Conference on Digital Libraries (JCDL), 2019, pp. 29–38.
[31] S. Prawiroatmodjo, Baasastra Jawa-Indonesia (Javanese-Indonesian Dictionary), Gunung Agung, Jakarta, 1981 (in Javanese and Bahasa Indonesia).
[32] K.F. Holle, Hamong Tani: de vriend van den Javannschen landman (Hamong Tani: the friend of the Javanese countryman), Landsdrukkerij, Batavia, 1876 (in Javanese, translated from Dutch by F. L. Winter).