J Inf Process Syst, Vol.18, No.4, pp.549~561, August 2022
https://doi.org/10.3745/JIPS.04.0250

ISSN 1976-913X (Print)
ISSN 2092-805X (Electronic)

JOURNAL OF INFORMATION PROCESSING SYSTEMS JIPS

Bi-directional Maximal Matching Algorithm to Segment Khmer Words in Sentence

Makara Mao[1], Sony Peng[1], Yixuan Yang[1], and Doo-Soon Park[2,*]

Abstract

In the Khmer writing system, the Khmer script is the official letter of Cambodia, written from left to right without a space separator; it is complicated and requires more analysis studies. Without clear standard guidelines, a space separator in the Khmer language is used inconsistently and informally to separate words in sentences. Therefore, a segmented method should be discussed with the combination of the future Khmer natural language processing (NLP) to define the appropriate rule for Khmer sentences. The critical process in NLP with the capability of extensive data language analysis necessitates applying in this scenario. One of the essential components in Khmer language processing is how to split the word into a series of sentences and count the words used in the sentences. Currently, Microsoft Word cannot count Khmer words correctly. So, this study presents a systematic library to segment Khmer phrases using the bi-directional maximal matching (BiMM) method to address these problematic constraints. In the BiMM algorithm, the paper focuses on the Bi-directional implementation of forward maximal matching (FMM) and backward maximal matching (BMM) to improve word segmentation accuracy. A digital or prefix tree of data structure algorithm, also known as a trie, enhances the segmentation accuracy procedure by finding the children of each word parent node. The accuracy of BiMM is higher than using FMM or BMM independently; moreover, the proposed approach improves dictionary structures and reduces the number of errors. The result of this study can reduce the error by 8.57% compared to FMM and BFF algorithms with 94,807 Khmer words.

Keywords

Bi-directional Maximal Matching, Khmer Language, Natural Language Processing, Word Corpus, Word Segmentation

1. Introduction

Khmer language is the official language for Cambodians (Khmer) to operate in terms of documentation, academic writing, and formal speaking. Khmer has been significantly affected by Sanskrit and Pali. In the writing system, the script is written from left to right. Each word within the same group of sentences mainly proceeds together without space between them. Usually, Spaces are vital utilized in the sentence to separate phrases, but they are not used to separate words in a phrase. The space separator in each sentence enables convenience and assistance for reading purposes [1]. The characters in the Cambodian script (called Khmer letter) formerly consisted of 35 consonants. However, only 33 consonants are used in the Cambodian writing system, which is arranged in five groups according to the position of the

Manuscript received September 9, 2021; first revision November 17, 2021; accepted November 29, 2021.
* **Corresponding Author:** Doo-Soon Park (parkds@sch.ac.kr)
[1] Dept. of Software Convergence, Soonchunhyang University, Asan, Korea (makaramao07@gmail.com, peng.sony61@gmail.com, yangyxuan621@gmail.com)
[2] Dept. of Computer Software Engineering, Soonchunhyang University, Asan, Korea (parkds@sch.ac.kr)

articulation. For each consonant, there is an inherent vowel: â /ɑː/ or ô /ɔː/; equivalently, each consonant is said to belong to the two series, including a-series or o-series.

Furthermore, besides the letter, there are 14 independent vowels (or special characters) and 23 vowel forms in the official writing system [2]. Therefore, each Khmer word contains a complicated structure and a challenging digital writing format. With so many possible consonants, sub-consonants, and vowel combinations, the structural formulation of Khmer words necessitates an appropriate strategy to manage and organize Cambodian's future system automation. Indeed, creating and combining the prepositions and conjunctions in multiple sentences might make it challenging to define appropriate characteristic lists for the smart system. In addition, the multi-placement of numbers and spaces within a punctuation mark creates a complicated format [3-5]. Without rules, spacing and numbering can be confusing. According to advanced academic studies and well-known researchers, natural language processing (NLP) can be used to solve human problems and current issues, especially when extracting complicated information from text and classifying deep issues [6]. Since there is an application called over-the-top (OTT) conducted by the applied NLP, including text analytical problems, language recognition, targeted advertising, question answering, etc. [7,8]. NLP is an effective functional operation for deploying word segmentation to many languages, including English, French, Chinese, Japanese, German, Russian, etc. [9-13]. However, the unsegmented words of the Khmer language are challenging and problematic for future NLP applications, including machine translation, speech recognition, information retrieval, or other application services [14-16]. Regarding these issues, this paper proposes a significant library to efficiently perform Khmer word segmentation to cope with these challenges and simplify further NLP system development in the Khmer language.

Word segmentation is one of the core components in NLP for partitioning each word into sentences or phrases. In this article, bi-directional maximal matching (BiMM) algorithm is a primary approach to resolve the existing difficulties in Khmer word segmentation. Furthermore, we use this algorithm to build an open library in sentences in a plain text format.

The rest of the paper is organized as follows. Section 2 presents the complementary related works, including a summary of the Khmer language overview, word pronunciations, diacritical marks, special characters, and Khmer word segmentation. Then, a thorough approach to word segmentation based on the applied BiMM algorithm is presented in Section 3. Finally, in Section 4, the conclusion and future work are given.

2. Related Work

2.1 Khmer Languages

Khmer script is the official letter for Cambodians to use daily, both for communication and academic purposes. To illustrate the detailed characters, Figs. 1–5 show the Khmer consonants and sub-consonants, numbers, independent vowels, dependent vowels, diacritical marks and special characters, respectively. Each letter consists of a particular function in a word structural formulation, which is highly complicated to apply segmentation.

ក ្ក[kâ]	ខ ្ខ[khâ]	គ ្គ[kô]	ឃ ្ឃ[khô]	ង ្ង[ngô]
ច ្ច[châ]	ឆ ្ឆ[chhâ]	ជ ្ជ[chô]	ឈ ្ឈ[chhô]	ញ ្ញ[nhô]
ដ ្ដ[dâ]	ឋ ្ឋ[thâ]	ឌ ្ឌ[dô]	ឍ ្ឍ[thô]	ណ ្ណ[nâ]
ត ្ត[tâ]	ថ ្ថ[thâ]	ទ ្ទ[tô]	ធ ្ធ[thô]	ន ្ន[nô]
ប ្ប[bâ]	ផ ្ផ[phâ]	ព ្ព[pô]	ភ ្ភ[phô]	ម ្ម[mô]
យ ្យ[yô]	រ ្រ[rô]	ល ្ល[lô]	វ ្វ[vô]	ឝ ្ឝ[shâ]
ឞ ្ឞ[ssô]	ស ្ស[sâ]	ហ ្ហ[hâ]	ឡ none[lâ]	អ ្អ[â]
The letters in yellow column are not use in now				

Fig. 1. Khmer consonants and consonant subscripts.

Khmer Numerals	០	១	២	៣	៤	៥	៦	៧	៨	៩
Arabic Numerals	0	1	2	3	4	5	6	7	8	9

Fig. 2. Khmer numbers.

ឥ [ĕ]	ឦ [ei]	ឧ [ŏ]	ឨ [ɔ]	ឩ [ŭ]
ឪ [âu]	ឫ [rœ̆]	ឬ [rœ]	ឭ [lœ̆]	ឮ [lœ]
ឯ [ê]	ឰ [ai]	ឱ, ឲ [aô]	ឳ [au]	

The letter in highlighted column is not currently used in modern Khmer language

Fig. 3. Khmer independent vowels.

ា [a]	ិ [ə]	ី [əj]	ឹ [ə]	ឺ [əɨ]
ុ [o]	ូ [ou]	ួ [uə]	ើ [aə]	ឿ [ɨə]
ៀ [iə]	េ [ei]	ែ [ae]	ៃ [aj]	ោ [ao]
ៅ [aw]	ុំ [om]	ំ [ɑm]	ាំ [am]	ះ [aʰ]
ិះ [eʰ]	ុះ [oʰ]	េះ [eʰ]	ោះ [ɑʰ]	

Fig. 4. Khmer dependent vowels.

ំ	ះ	ៈ	៉	៊	ុ	់	៌	៍	៏	៎
័	៑	្	ៗ	៘	។	៕	៖	?	!	.
″	"	«	»	៙	៚					

Fig. 5. Khmer diacritical marks and other special characters.

2.2 Khmer Word Segmentation

Based on previous BiMM implementation for plain text and Microsoft Word documents, the scope of this algorithm has improved and deployed in various languages [16-18]. The BiMM algorithm with Khmer clusters for dictionary lookup is used to encounter adequate words in the defined containers. StringBuilder and string interning is used for reducing memory usage and accelerating string search queries in string pools [19]. Based on these techniques, the performance evaluation, conducted on 160,000 Khmer words in plain text documents, resulted in up to 2.58 seconds of searching queries and 98.13% accuracy metric [20]. Khmer word segmentation used conditional random fields and achieved an average F1-score of 0.99 and 0.92 on testing data and maximal matching based on achieving an average

F1-score, respectively [21]. Khmer spelling checker is an approach based on an n-gram hidden Markov model (HMM) and C4.5 algorithm to achieve an accuracy metric of 94% with word correction features [22]. The paper on Khmer word segmentation with maximum matching uses clustering to define a rule and match all the Khmer clusters. The lookup procedure checks all words from the corpus to perform the segmentation [23].

3. Bi-directional Maximal Matching for Word Segmentation

This part describes algorithm designs and flows processing for Khmer word segmentation in NLP. The sub-elements of the proposed BiMM approach consist of five primary phases, including maximal matching methods, Khmer clusters, digital tree or prefix tree, word corpus, and algorithm flows.

The maximal matching algorithm is an optimal strategy for precisely segmenting Khmer words. Forward maximal matching (FMM) and backward maximal matching (BMM) are enablers for the proposed BiMM approach. With sole FMM or BMM, the word segmentation errors are feasibly higher than in other conditions. In addition, the proposed approach selects an optimal matching method in a suitable condition with sufficient comparison and recommendation methods by minimizing the error possibilities. FMM and BMM execute opposite directions to formulate every word counting possibility. The proposed approach targets a better precision than traditional positive and reverses maximal matching algorithms. The processing flow of BiMM for Khmer word segmentation in a sentence is illustrated in Fig. 6.

FMM algorithm works with the corpus lists and identically operates based on dictionary algorithms.

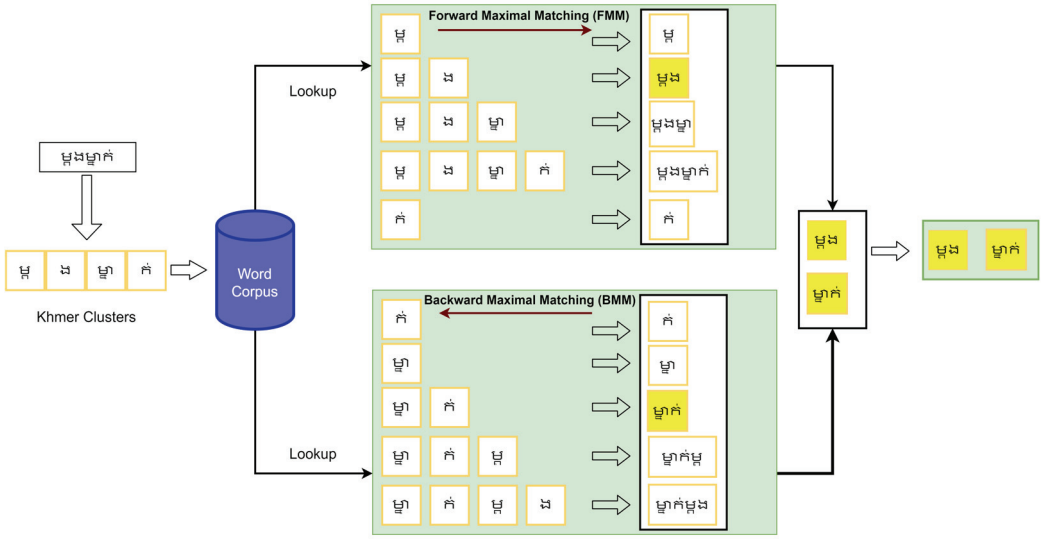

Fig. 6. Processing flow of BiMM for Khmer word segmentation in a sentence.

This approach priorities the corpus list with maximal letter counts in a forward matching direction. If a word is found, FMM algorithm makes a boundary at the end of the word with maximal letter counts, then the algorithm continues to search for the next word with same processing flows. BMM algorithm generates word boundaries by backward comparison between FMM, BMM, and BiMM are shown in Fig.

7. BiMM algorithm combined FMM with BMM to determine the optimal word segmentation in different conditions. In addition, BiMM shares the similarity with both FMM and BMM, which executes with both direction in parallel manner. As shown in Fig. 7, the differentiation between BMM, FMM, and BiMM presents the error possibilities of each method following by different conditional sentences. Procedure steps define the thorough processes of BiMM as loading words from the corpus list to store in the hash table, generating Khmer clusters of inputted unsegmented text, and looping through the Khmer clusters list (Fig. 8).

Forward Maximal Matching (FMM)
ម្តងម្នាក់ ⇨ ម្តងម្នា + ក់ (Incorrect) សម្លចាស់អំបិលមានរសជាតិឆ្ងាញ់ ⇨ សម្ល + ចាស់ + អំបិល + មាន + រសជាតិ + ឆ្ងាញ់
Backward Maximal Matching (BMM)
ម្តងម្នាក់ ⇨ ម្តង + ម្នាក់ ⇨ សម្លចាស់អំបិលមានរសជាតិឆ្ងាញ់ ⇨ សម្ល + ចាស + ់ + អំបិល + មាន + រសជាតិ + ឆ្ងាញ់ (Incorrect)
Bi-directional Maximal Matching(BiMM)
ម្តងម្នាក់ ⇨ ម្តង + ម្នាក់ សម្លចាស់អំបិលមានរសជាតិឆ្ងាញ់ ⇨ សម្ល + ចាស់ + អំបិល + មាន + រសជាតិ + ឆ្ងាញ់

Fig. 7. Comparison between FMM, BMM, and BiMM.

Words	Number of clusters	Clusters
ខ្ញុំ	1	[ខ្ញុំ]
អ្នក	2	[អ្ន] + [ក]
អ្នកគ្រូ	3	[អ្ន] + [ក] + [គ្រូ]
អារម្មណ៍	4	[អា] + [រ] + [ម្ម] + [ណ៍]

Fig. 8. Khmer clusters.

A digital tree or prefix tree is also known as a trie, a search tree as strings. A trie can be used to find children's nodes of the parents' nodes in the dictionary, comparable to matching algorithms. We use trie to search words as the string to accelerate rapid searching queries on a binary tree. The process of trie data in Khmer word segmentation from binary data finds the root node for a word from one node to another. The example words are provided as follows, including "អ្នកគ្រូ," "អង្គរ," and "ភាសាខ្មែរ" (Fig. 9).

The processed for writing Khmer script in Khmer Unicode spelling order are presented as follows.

- អ + ្ន + ក + ា + ្រ + ូ = អ្នកគ្រូ (Female teacher)
- អ + ង + ្គ + រ = អង្គរ (Angkor)
- ភ + ា + ស + ា + ខ + ្ម + ែ + រ = ភាសាខ្មែរ (Khmer language)

An illustrate detail on the word "អ្នកគ្រូ" follow on Khmer dictionary spelling rule; the procedures are defined as follows:

- Consonant អ/'â /
- ◌្ (char\u17d2) to convert a consonant to a sub-consonant
- So, the consonant ន/nô/ become the sub-consonant ◌្ន (◌្ + ន = ◌្ន)/jerng nô /
- Spelled with consonant ក/kâ /
- Spelled with consonant ក/kô /
- ◌្ (char\u17d2) to convert a consonant to a sub-consonant
- So, the consonant រ/rô/ become the sub-consonant ◌្រ (◌្ + រ = ◌្រ)/jerng rô/
- The final vowel ◌ូ/ou/ is muted.

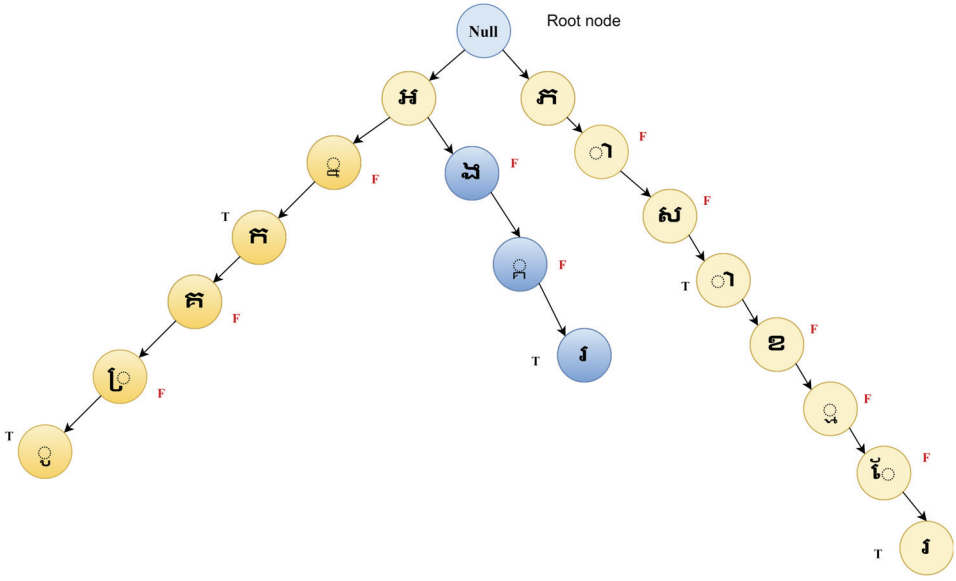

Fig. 9. Trie data searching structure from word corpus.

Since Khmer word segmentation is proposed, word corpus is a significant feature for word segmentation from sentence to word as a container. The word corpus that is used in this library contains four primary principles as follows:

- All words in the corpus are not duplicated.
- Order alphabets do not require all words in the corpus.
- From one word to another are separated by a newline (“\n”).
- All words and numbers are separated by tab (“\t”).

Word corpus plays an essential role in Khmer word segmentation, with totals of 94,806 gathered from websites, news, and academic documents. Pseudocode algorithms for non-Khmer-word detection, inclu-

ding numbers and English, and Khmer word segmentation are presented in Algorithms 1 and 2, respectively. Algorithm 1 describes the pseudocode of non-Khmer-word detection of user inputs. The method flow illustrates four primary functions, including number, parseNumber, English, and parseEnglish, that handle whether the input text is a numerical value, numerical combination, English characters, and English words

Algorithm 1. Pseudocode for non-Khmer-word detection (numbers and English)

Require: input sentence i, OpenLibrary package
Ensure: non-Khmer word detection on i
define number (i) **do**
    check characters in "0123456789๐๑๒๓๔๕๖๗๘๙"
define parseNumber ($index$) **do**
    declare result lists
    while (index < length of text) **do**
      if number(i) is true **do**
          append the result with text[index]
      else
          return result
      end if
    end while
define english (i) **do**
    check characters Both uppercase and lowercase of English a-z and A-Z
define parseEnglish ($index$) **do**
    declare result lists
    while (index < length of text) **do**
      if english (i) is true **do**
          append the result with text[index]
      else
          return result
      end if
    end while
  end procedure

Algorithm 2 shows the pseudocode procedure of Khmer word segmentation, which includes two main functions: trie and check words. The trie function determines the process of word-level detection, whether the input text, denoted as *i*, is the child of a prefix, suffix, or ordinary word combination. Word and found word containers are declared to store the segmented words if it is found in the corpus. While the index is smaller than the length of *i*, the child container obtains the input text values of that particular sequential index. The child values are appended to the word container correspondingly. However, if the word is found within the predefined prefix package, denoted as prefix-list, the model will append that particular word to the foundWord container. And, if the word is an ordinary Khmer word, the model will return that particular word directly. Otherwise, the model returns the existing foundWord container. The next index will proceed accordingly after the abovementioned condition is completed.

Algorithm 2. Pseudocode for segmenting Khmer word

```
define trie (i, index) do
    declare container word and foundWord
    while (index < length of i) do
      child = inputText[index]
      append child to word container
      if the word is found in prefixList do
          append a word to foundWord container
      else if the word is a regular word do
          return word
      else
          return foundWord container
      end if
       index +=1
    end while
end Trie procedure
define checkWords do
    declare temp container
    while (startIndex < length of i) do
          child = i [startIndex]
          declare word container
          if the split word isNumber do
             word=parseNumber(startIndex) encode to 'utf-8'
          else if the split word isEnglish do
             word=parseEnglish(startIndex) encode to 'utf-8'
          else
             word=parseTrie (startIndex)
          end if
          count the word length and store in length list
          if length=0 do
             append child to temp container
             startIndex += 1
             if 'startIndex >= length of i' or 'length of temp>0' do
                append the temp to the result container
                re-declare empty temp
             end if
          end if
          decode the word to 'utf-8' and append to result
          startIndex+=length
      end while
      return 'segmented words from result' and 'length of result words'
   end checkWords procedure
```

CheckWords function determines the attainable compound words of the user input text by firstly declaring the temporary segmented words container, denoted as a temp. While the starting index, denoted as startIndex, is smaller than the length of i, the system appends the input text of a particular initialled startIndex towards the child container. If the split words are numerical values or English characters identified in algorithm 1, the model will gather parseNumber or parse English of startIndex to word container, respectively. Otherwise, the split words will process by parseTrie function. To fully support the Khmer word characters, UTF-8 encoding is used. Every word length is counted and stored in a length list. If the length is empty, the model will append the overall child to the temp container, and the startIndex will be increased by 1. If the startIndex is greater than/equal to the length of i or the length of temp is greater than 0, the existing temp will be appended to the defined final result container. After appending, a new empty temp is re-defined. The utf-8 decoding procedure is configured to support the Khmer word characters, which are later appended to the result container. The startIndex is correspondingly increased by the length values.

4. Results and Discussion

In the simulated model, Python programming is mainly conducted to develop the system. The experimental data are gathered from Khmer documentation. Nineteen sentences were used to input in the algorithm. Each sentence contains the containers for result, error, count word, and startIndex. The proposed system will check both BMM and FMM. Within BMM procedures, the input sentence is reversed backwards; otherwise, the input text processes forward direction. While the startIndex is between the length of input text, each word is partitioned as ch. By using the three functions, including parseNumber, parseEnglish, and parseTrie, the checking procedures of ch are precise. The error words are captured and appended to the final outputs. Fig. 10 presents the flowchart of the experimental environment within the proposed scheme. len is denoted as the length of the variable in this scheme. The final output optimized the maximal matching procedures and reduced a significant amount of error word segmentation. Following the proposed approach procedures, the system is finalized with notable outperformance compared to FMM and BMM. The final outputs display the user original i, segmented words separation from the result container, and the total length of words, which is illustrated in Fig. 11.

The processes of implementation of BiMM of Khmer word segmentation in the sentence are described as follows. Firstly, the procedure is to input the sentences in the list and load the open library package, then load the corpus list are made to generate words from the corpus into the Hash table. Khmer clusters from the inputted unsegmented text are generated and looped through the list. Secondly, before processing word segmentation, we need to clean up all word that has space ("\u200B") between each word in the sentence and set a limitation from the starting sentence until the ending. Thirdly, based on the Khmer cluster and Khmer Unicode spelling, words are segmented into Khmer clusters and find the correct word one by one from the corpus list match with the document.

Furthermore, the segmentation process is based on BiMM with the dictionary lookup using the Hash table. Then, the unknown word and error word detection is processing and executed to store word in a list. In the final step, after acquiring the result from the previous steps, a defined word from dictionary lookup starts to segment words in the sentence and count the total words that follow the defined rules. Fig. 12 presents the result comparison between FMM, BMM, and BiMM. Within 19 input sentences, the

error reduction achieved an 8.57% improvement over FMM and BMM algorithms. The total error counts between FMM, BMM, and BiMM are 70, 70, and 64, respectively.

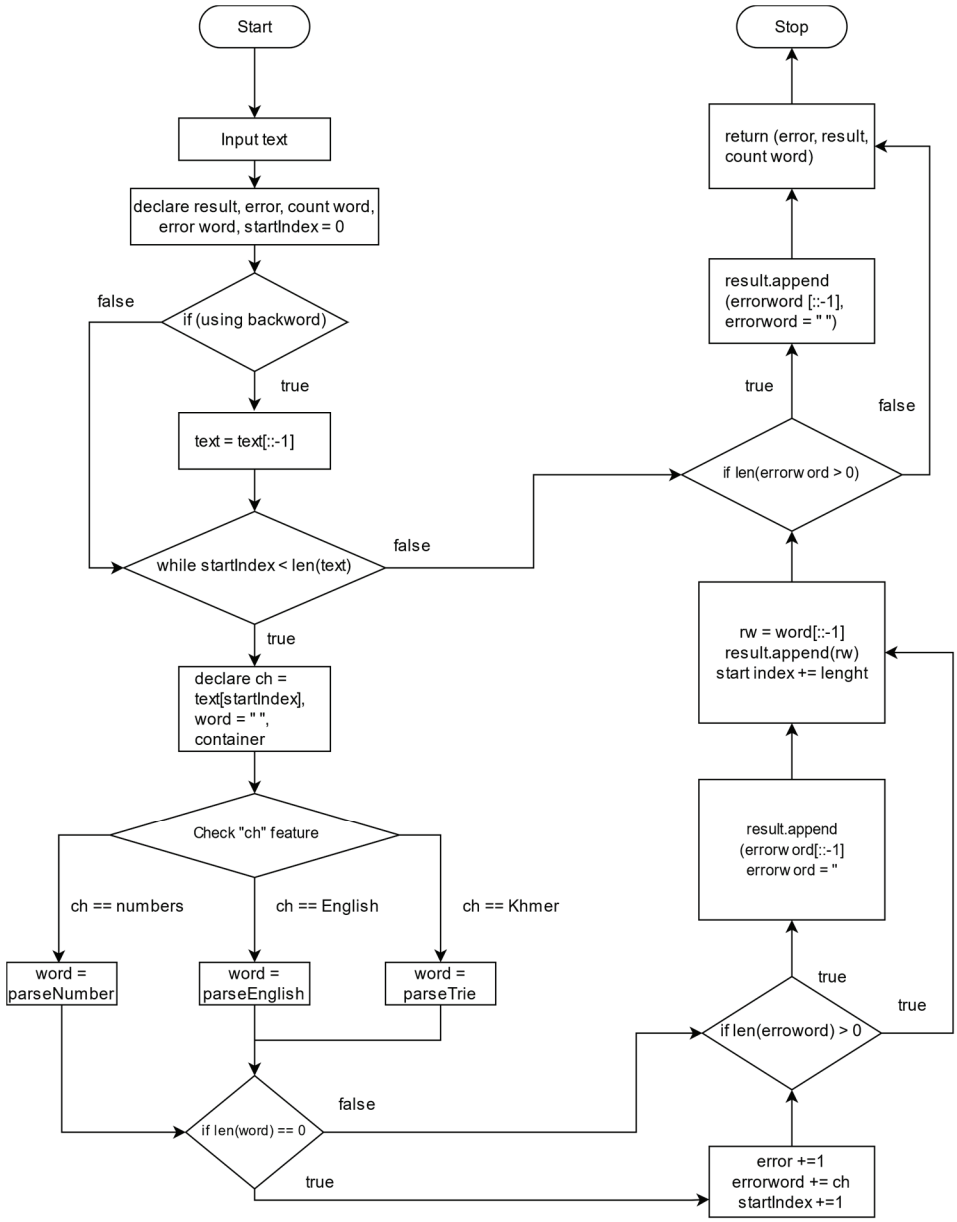

Fig. 10. Flowchart of experimental environment implementation.

Original Text	ទឹកត្រីដែលឆ្ងាញ់ គឺធ្វើមកពីត្រីដែលរស់នៅក្នុងទឹកប្រៃ
Segment	['ទឹកត្រី', 'ដែល', 'ឆ្ងាញ់', 'គឺ', 'ធ្វើ', 'មកពី', 'ត្រី', 'ដែល', 'រស់នៅ', 'ក្នុង', 'ទឹកប្រៃ']
Total Words	11

Fig. 11. The final outputs of BiMM Khmer word segmentation results.

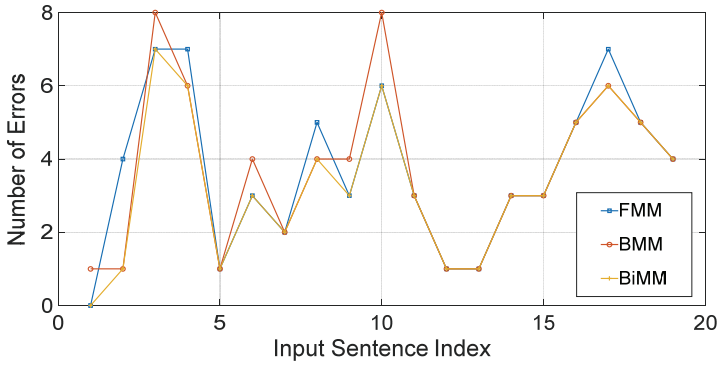

Fig. 12. Result comparison between FMM, BMM, and BiMM.

5. Conclusion

This research aims to improve Khmer word segmentation based on the BiMM algorithm. The paper tackles challenging problems in Khmer sentence construction by using the proposed BiMM algorithm. Splitting Khmer words, counting the segmented words, and multi-direction detection were presented. Digital or prefix tree optimally discovering the child nodes from the word lookup were deployed to enhance the model performance. Khmer clusters were used to generate Khmer characters to reach an acceptable accuracy from multiple Khmer writing possibilities. Corpus words were created to improve word segmentation accuracy. In future studies, Khmer words extension in corpus lists, misspelling words, unknown words, and phrases in sentences will be considered for system enhancement.

Acknowledgement

This research was supported by the National Research Foundation of Korea (No. NRF-2020RIA2B5B01002134) and the BK21 FOUR (Fostering Outstanding Universities for Research; No. 5199990914048).

References

[1] C. Ding, M. Utiyama, and E. Sumita, "NOVA: a feasible and flexible annotation system for joint tokenization and part-of-speech tagging," *ACM Transactions on Asian and Low-Resource Language Information Processing*, vol. 18, no. 2, article no. 17, 2019. https://doi.org/10.1145/3276773

[2] R. Buoy, S. Kor, and N. Taing, "An end-to-end Khmer optical character recognition using sequence-to-sequence with attention," 2021 [Online]. Available: https://arxiv.org/abs/2106.10875.

[3] X. Yan, X. Xiong, X. Cheng, Y. Huang, H. Zhu, and F. Hu, "HMM-BiMM: hidden Markov model-based word segmentation via improved bi-directional maximal matching algorithm," *Computers & Electrical Engineering*, vol. 94, article no. 107354, 2021. https://doi.org/10.1016/j.compeleceng.2021.107354

[4] M. Sassano, "Deterministic word segmentation using maximum matching with fully lexicalized rules," in *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL), Volume 2: Short Papers*, Gothenburg, Sweden, 2014, pp. 79-83.

[5] C. Ding, Y. K. Thu, M. Utiyama, and E. Sumita, "Word segmentation for Burmese (Myanmar)," *ACM Transactions on Asian and Low-Resource Language Information Processing*, vol. 15, no. 4, article no. 22, 2016. https://doi.org/10.1145/2846095

[6] J. M. Nobel, S. Puts, J. Weiss, H. J. Aerts, R. H. Mak, S. G. Robben, and A. L. Dekker, "T-staging pulmonary oncology from radiological reports using natural language processing: translating into a multi-language setting," *Insights into Imaging*, vol. 12, article no. 77, 2021. https://doi.org/10.1186/s13244-021-01018-1

[7] S. Liang, K. Stockinger, T. M. de Farias, M. Anisimova, and M. Gil, "Querying knowledge graphs in natural language," *Journal of Big Data*, vol. 8, article no. 3, 2021. https://doi.org/10.1186/s40537-020-00383-w

[8] D. Cao, X. Ren, M. Zhu, and W. Song, "Visual question answering research on multi-layer attention mechanism based on image target features," *Human-centric Computing and Information Sciences*, vol. 11, article no. 11, 2021. https://doi.org/10.22967/HCIS.2021.11.011

[9] M. Kuzma and A. Moscicka, "Evaluation of metadata describing topographic maps in a National Library," *Heritage Science*, vol. 8, article no. 113, 2020. https://doi.org/10.1186/s40494-020-00455-3

[10] H. Christian, D. Suhartono, A. Chowanda, and K. Z. Zamli, "Text based personality prediction from multiple social media data sources using pre-trained language model and model averaging," *Journal of Big Data*, vol. 8, article no. 68, 2021. https://doi.org/10.1186/s40537-021-00459-1

[11] H. Kamper, A. Jansen, and S. Goldwater, "Unsupervised word segmentation and lexicon discovery using acoustic word embeddings," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 4, pp. 669-679, 2016.

[12] C. Shorten, T. M. Khoshgoftaar, and B. Furht, "Text data augmentation for deep learning," *Journal of Big Data*, vol. 8, article no. 101, 2021. https://doi.org/10.1186/s40537-021-00492-0

[13] R. Buoy, N. Taing, and S. Kor, "Joint Khmer word segmentation and part-of-speech tagging using deep learning," 2021 [Online]. Available: https://arxiv.org/abs/2103.16801.

[14] K. M. Park, H. C. Cho, and H. C. Rim, "Utilizing various natural language processing techniques for biomedical interaction extraction," *Journal of Information Processing Systems*, vol. 7, no. 3, pp. 459-472, 2011.

[15] K. Batsuren, E. Batbaatar, T. Munkhdalai, M. Li, O. E. Namsrai, and K. H. Ryu, "A dependency graph-based keyphrase extraction method using anti-patterns," *Journal of Information Processing Systems*, vol. 14, no. 5, pp. 1254-1271, 2018.

[16] V. Chea, Y. K. Thu, C. Ding, M. Utiyama, A. Finch, and E. Sumita, "Khmer word segmentation using conditional random fields," in *Proceedings of the 2nd Annual Conference on Khmer Natural Language Processing (KNLP)*, Phnom Penh, Cambodia, 2015, pp. 62-69.

[17] D. Li, J. Wang, M. Chen, Z. Zhang, and Z. Li, "Base-band involved integrative modeling for studying the transmission characteristics of wireless link in railway environment," *EURASIP Journal on Wireless Communications and Networking*, vol. 2015, article no. 81, 2015. https://doi.org/10.1186/s13638-015-0316-3

[18] F. N. A. Al Omran and C. Treude, "Choosing an NLP library for analyzing software documentation: a systematic literature review and a series of experiments," in *Proceedings of 2017 IEEE/ACM 14th International Conference on Mining Software Repositories (MSR)*, Buenos Aires, Argentina, 2017, pp. 187-197.

[19] S. Knight, *NLP at Work: The Difference that Makes the Difference*, 4th ed. London, UK: Nicholas Brealey Publishing, 2020.

[20] N. Bi and N. Taing, "Khmer word segmentation based on bi-directional maximal matching for plaintext and Microsoft Word document," in *Proceedings of 2014 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)*, Siem Reap, Cambodia, 2014, pp. 1-9.

[21] S. Kundu and G. Sarker, "A multi-level integrator with programming based boosting for person authentication using different biometrics," *Journal of Information Processing Systems*, vol. 14, no. 5, pp. 1114-1135, 2018.

[22] P. Hok, "Khmer Spell Checker," M.S. thesis, Australian National University, Canberra, Australia, 2005.

[23] S. Chea, M. Soeurn, S. Kor, and S. Srun, "Khmer word segmentation with Maximum Matching," in *Proceedings of the 10th International Conference on Internet (ICONI)*, Phnom Penh, Cambodia, 2018.

Makara Mao **https://orcid.org/0000-0003-3321-6716**

He received B.S. degree in computer science from Royal University of Phnom Penh, Cambodia in 2016. He is currently studying (Combined Master's and Ph.D. degrees) at Soonchunhyang University, South Korea. His current research includes Natural Language Processing (NLP), deep learning and data mining.

Sony Peng **https://orcid.org/0000-0003-3847-9662**

She received B.S. degree in IT engineering from Royal University of Phnom Penh, Cambodia, in 2018. She is currently a M.S. candidate at Soonchunhyang University, South Korea. Her research interests include data mining, mobile AI, and cloud computing.

Yixuan Yang **https://orcid.org/0000-0002-9334-566X**

She received the B.Sc. degree in Software Engineering from Taiyuan University of Technology, China, in 2017 and the M.Sc. degree in Software Engineering from Shaanxi Normal University, China, in 2020. She is currently pursuing her Ph.D. degree at Soonchunhyang University, South Korea. Her research interests include social computing, community detection and formal concept analysis.

Doo-Soon Park **https://orcid.org/0000-0002-2776-8832**

He received his Ph.D. in Computer Science from Korea University in 1988. Currently, he is a professor in the Department of Computer Software Engineering at Soonchunhyang University, South Korea. He is Dean of Graduate School at Soonchunhyang University, Director of BK21 FOUR Well-Life Big Data at Soonchunhyang University, Director of Wellness Service Coaching Center at Soonchunhyang University, and Director of Computer Software Research Group in KIPS. He was President of KIPS (Korea Information Processing Society) from 2015 to 2015, and Director of Central Library at Soonchunhyang University from 2014 to 2015. He was editor in chief of JIPS (*Journal of Information Processing Systems*) at KIPS from 2009 to 2012, and Dean of the Engineering College at Soonchunhyang University from 2002 to 2003. He has served as an organizing committee member of international conferences including, BIC 2021, MUE 2021, WorldIT 2021, CSA 2020, BIC 2019, FutureTech 2019, WorldIT 2019, FutureTech 2018, BIC 2018. His research interests include data mining, big data processing and parallel processing. He is a member of IEEE, ACM, KIPS, KMS, and KIISE.